

Industry Grade Project -1

Divya Devi V

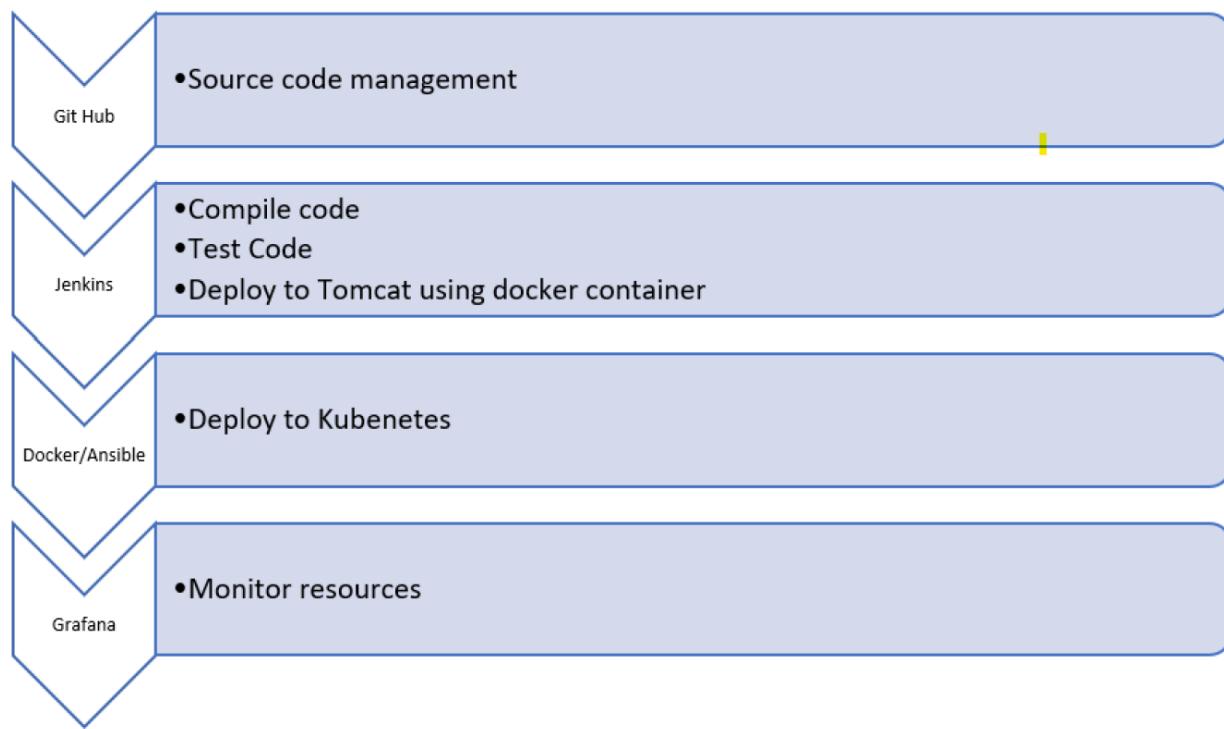
14-May-2023

Devops Certification

Edureka

Building a CI-CD pipeline

The Process Flow to be followed to achieve this.



Explaining the Tasks that will be done.

- Push the code to our GitHub repository.
- Create a continuous integration pipeline using Jenkins to compile, test, and package the code
- Write Dockerfile to push the war file to the Tomcat server.
- Integrate Docker with Ansible and write the playbook.
- Deploy artifacts to the Kubernetes cluster
- Monitor resources using Grafana.

What Softwares will be installed?

1. Java
2. Maven
3. Git
4. Jenkins
5. Docker
6. Ansible
7. Kubernetes
8. Grafana
9. Prometheus

Resources Required for these

1. AWS Account
2. MOBA-Xterm/Putty
3. Github account
4. Docker-hub account

We will also add details on the same document on how these installations and resource setups were done.

Let us get started on the tasks one by one.

Task 1: Clone the project from the GitHub link shared in resources to your local machine. Build the code using Maven commands.

Also here am going to do the git and maven installation on AWS EC2 server.

Steps done to complete the task.

1. Creating a git hub account.
2. Clone the project from Edureka resources to your IDE.
3. Installing git bash client and push the code to your github account
4. Verify the status of the push.
5. Installing maven
6. Build the code with maven



What is GIT?

Git is free and open-source software for distributed version control, tracking changes in any set of files, usually used for coordinating work among programmers collaboratively developing source code during software development.

Creating a GIT hub account

- Go to <https://github.com/join> to create a GitHub account.
- You'll need to verify your email during the signup process.
- Select the free plan account and create one.

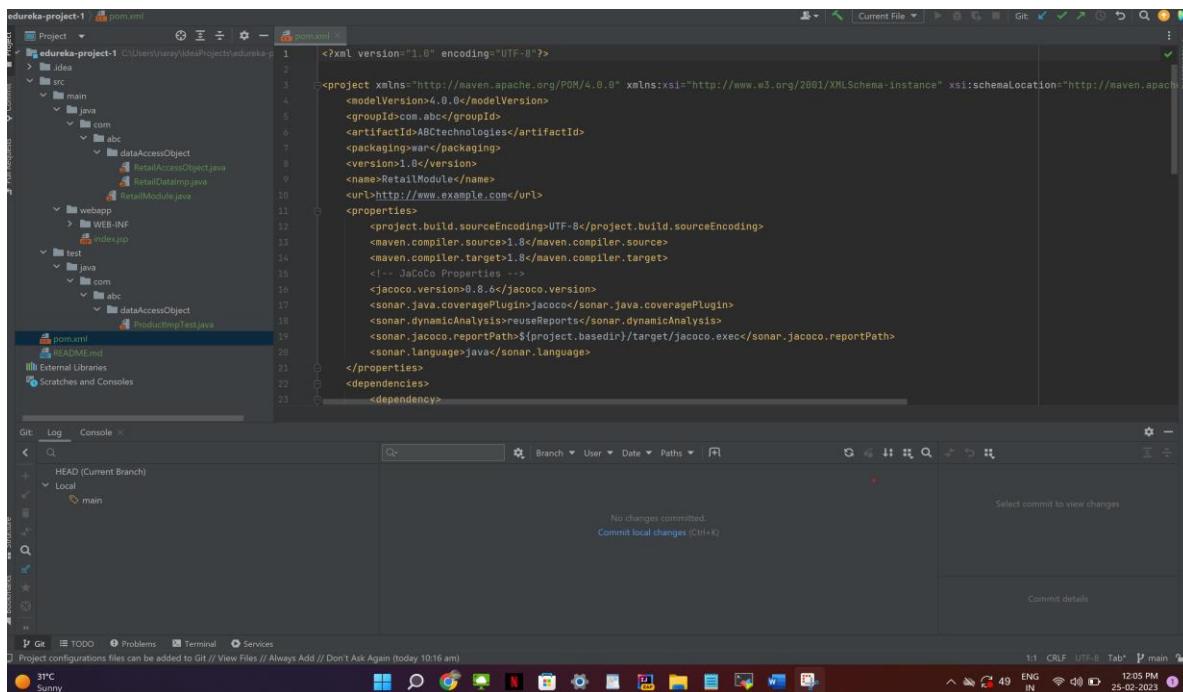
Successfully created github account.

Also created a initial repository with only a readme file.

<https://github.com/divyadevivasudevan/edureka-project-1.git>

Installed Git bash on my PC. Git BASH:: Git for Windows provides a BASH emulation used to run Git from the command line which is same as the linux env.

The git client is then used from our IDE -IntelliJ in my case. Copied the project from Edureka to my local

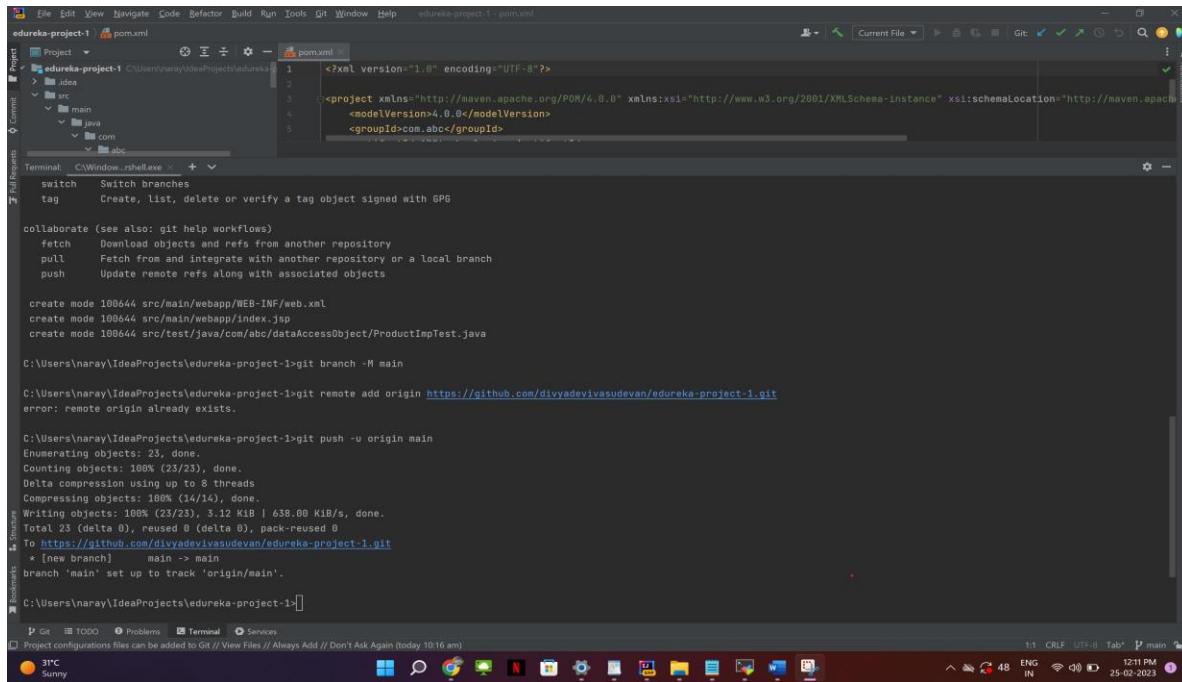


Committed and pushed the repo to my github remote repo using the git commands.
Git commands executed are:

- git init for initializing a local repository
- git add . to add all your files that the local repository
- git commit -m 'commit message' to save the changes you made to those files

- git remote add origin <https://github.com/divyadevivasudevan/edureka-project-1.git>
- git branch -M main (creating a branch main)
- git push -u origin main – This pushes my code in the local to the main branch in the remote.

The Git logs can be seen..



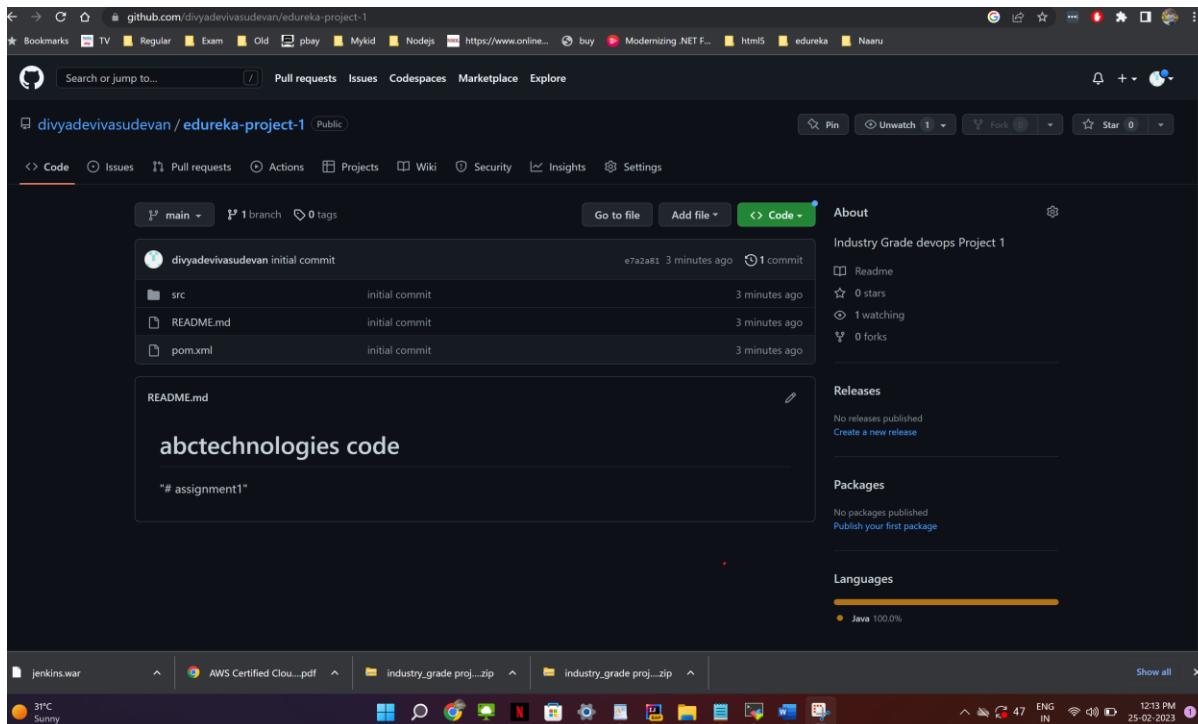
```

edureka-project-1 pom.xml
edureka-project-1 C:\Users\naray\IdeaProjects\edureka-project-1>git branch -M main
C:\Users\naray\IdeaProjects\edureka-project-1>git remote add origin https://github.com/divyadevivasudevan/edureka-project-1.git
error: remote origin already exists.

C:\Users\naray\IdeaProjects\edureka-project-1>git push -u origin main
To https://github.com/divyadevivasudevan/edureka-project-1.git
 * [new branch]  main -> main
branch 'main' set up to track 'origin/main'.
C:\Users\naray\IdeaProjects\edureka-project-1>

```

Verified the Github whether commit and push was successful.



github.com/divyadevivasudevan/edureka-project-1

divyadevivasudevan / edureka-project-1 Public

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main 1 branch 0 tags

About

Industry Grade devops Project 1

Readme 0 stars 1 watching 0 forks

Releases

No releases published Create a new release

Packages

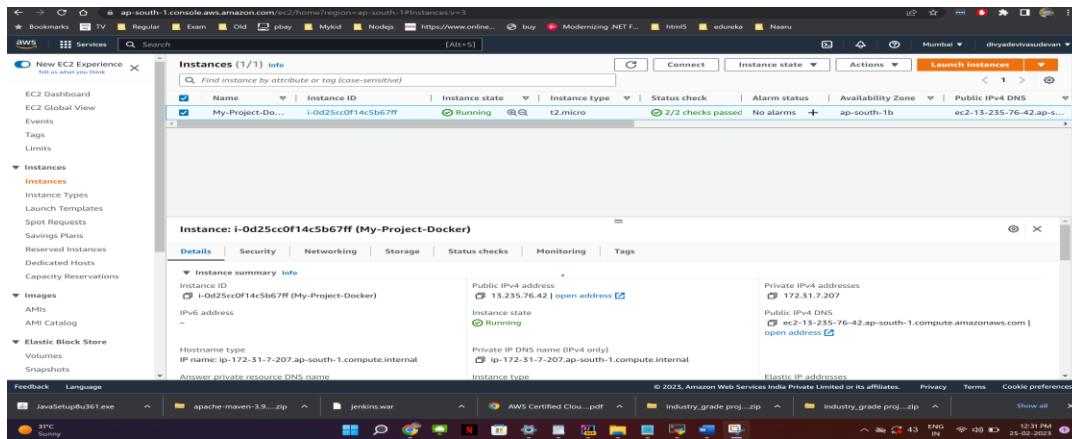
No packages published Publish your first package

Languages

Java 100.0%

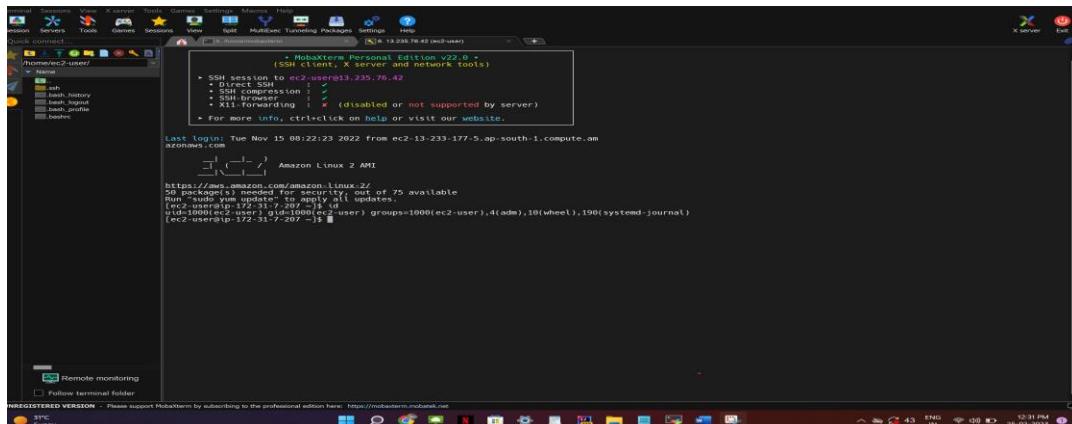
jenkins.war AWS Certified Cloud...pdf industry_grade proj...zip industry_grade proj...zip

Launch an EC2 instance.

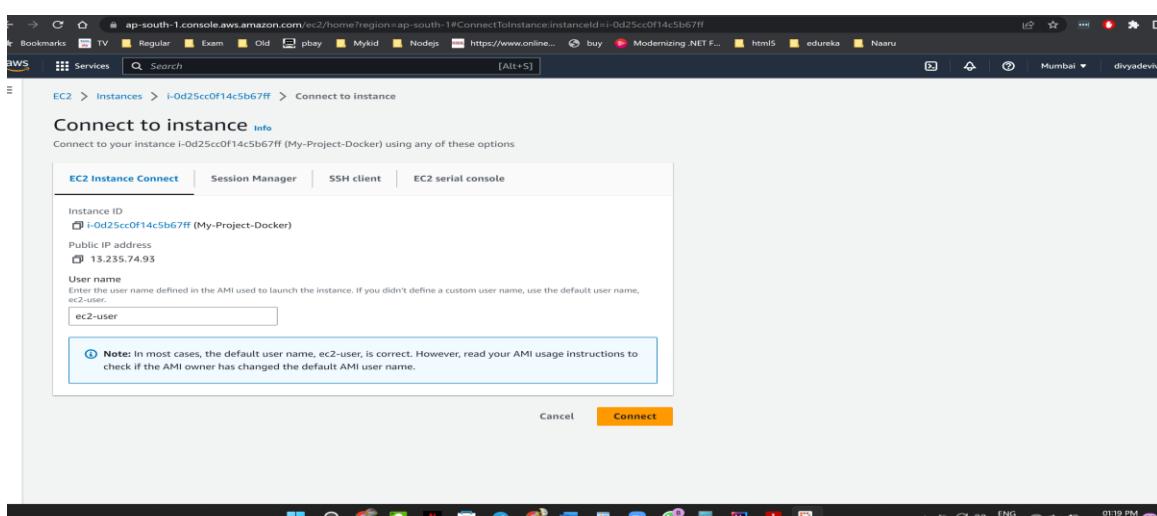


Connect to the instance with moba-xtrem or any ssh client using the PEM.

Moba-xtrem:



Can also use the connect ssh client facility available in the EC2 portal which is much simpler in process.



Command to verify the OS version : cat /etc/os-release.

```
[ec2-user@ip-172-31-7-207 ~]$ cat /etc/os-release
NAME="Amazon Linux 2"
VERSION="2.0.2023.05.05.1"
ID="amzn"
ID_LIKE="centos"
PRETTY_NAME="Amazon Linux 2"
CPE_NAME="cpe:2.3:amazon:amazonlinux:2"
[ec2-user@ip-172-31-7-207 ~]$
```

Installing GIT and MAVEN and Run MVN commands

Step 1:: Install Git -> Command : sudo yum install git -y

Step 2::clone the repo in ec2 ->Command – git clone<https://github.com/divyadevivasudevan/edureka-project-1.git>

Committed the installation script to git repo – file name git/installgit.sh

```
[ec2-user@ip-172-31-7-207 ~]$ sudo yum install git -y
[sudo] password for ec2-user:
Last metadata expiration check: 0:00:00 ago on Mon May 15 01:10:28 2023.
0:00:00
[ec2-user@ip-172-31-7-207 ~]$
```

```
[ec2-user@ip-172-31-7-207 ~]$ git clone https://github.com/divyadevivasudevan/edureka-project-1.git
Cloning into 'edureka-project-1'...
remote: Enumerating objects: 27, done.
remote: Counting objects: 100% (27/27), done.
remote: Compressing objects: 100% (17/17), done.
remote: Total 27 (delta 1), reused 27 (delta 1), pack-reused 0
Receiving objects: 100% (27/27), done.
Resolving deltas: 100% (1/1), done.
[ec2-user@ip-172-31-7-207 ~]$
```

Maven™ What is Maven?

Maven is a build automation tool used primarily for Java projects. Maven can also be used to build and manage projects written in C#, Ruby, Scala, and other languages.

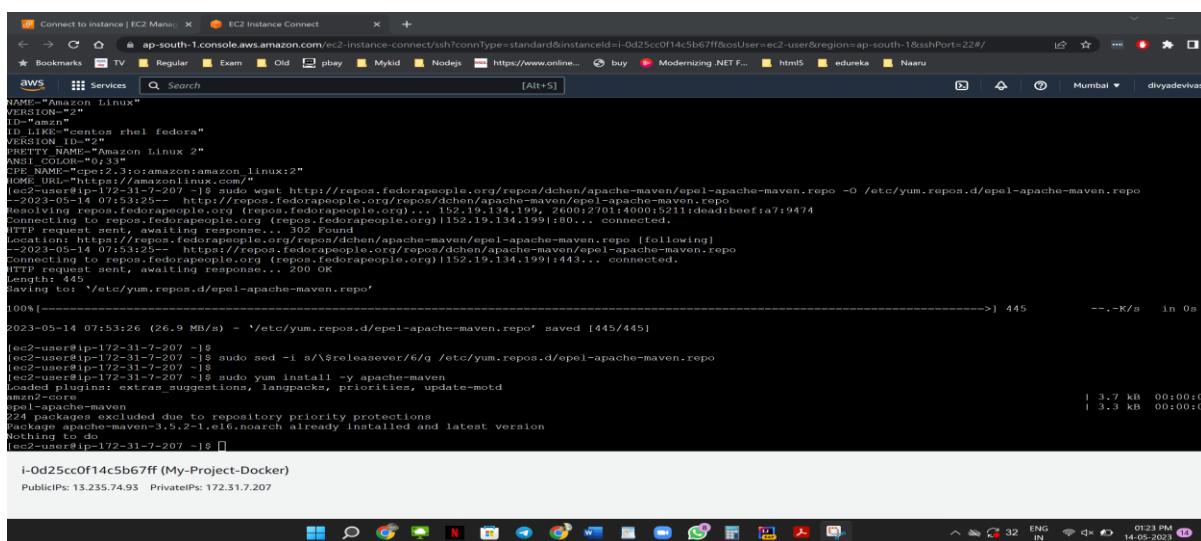
#Install maven in the server

```
sudo wget http://repos.fedorapeople.org/repos/dchen/apache-maven/epel-apache-maven.repo -O /etc/yum.repos.d/epel-apache-maven.repo
```

```
sudo sed -i s/\$releasever/6/g /etc/yum.repos.d/epel-apache-maven.repo
```

```
sudo yum install -y apache-maven
```

Committed the installation script to git repo – file name maven/installmaven.sh



```
NAME="Amazon Linux"
VERSION="2"
ID="amzn"
ID_LIKE="centos_rhel fedora"
VERSION_ID="2"
PRETTY_NAME="Amazon Linux 2"
ANSI_COLOR="#339933"
CPE_NAME="cpe:2.3:r:amazon:amazon_linux:2"
HOME_URL="https://amazonlinux.amazon.com"
[ec2-user@ip-172-31-7-207 ~]$ sudo wget http://repos.fedorapeople.org/repos/dchen/apache-maven/epel-apache-maven.repo -O /etc/yum.repos.d/epel-apache-maven.repo
2023-05-14 07:53:25 [ec2-user@ip-172-31-7-207 ~]$ sudo sed -i s/\$releasever/6/g /etc/yum.repos.d/epel-apache-maven.repo
[ec2-user@ip-172-31-7-207 ~]$ sudo yum install -y apache-maven
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core
apache-maven
apache-maven-extras excluded due to repository priority protections
Package apache-maven-3.5.2-1.el6.noarch already installed and latest version
Nothing to do
[ec2-user@ip-172-31-7-207 ~]$ [ec2-user@ip-172-31-7-207 ~]$ i-0d25cc0f14c5b67ff (My-Project-Docker)
PublicIPs: 13.235.74.93 PrivateIPs: 172.31.7.207
[ec2-user@ip-172-31-7-207 ~]$
```

Basic Maven Commands :

- 1) mvn --version to find the version of the maven install

```
[ec2-user@ip-172-31-7-207 ~]$ mvn --version
Apache Maven 3.5.2 (138edd61fd100ec658bfa2d307c43b76940a5d7d; 2017-10-18T07:58:13Z)
Maven home: /usr/share/apache-maven
Java version: 17.0.6, vendor: Amazon.com Inc.
Java home: /usr/lib/jvm/java-17-amazon-corretto.x86_64
Default locale: en_US, platform encoding: UTF-8
OS name: "linux", version: "5.10.167+147.601.amzn2.x86_64", arch: "amd64", family: "unix"
[ec2-user@ip-172-31-7-207 ~]$
```

- ```
i-0d25cc0f14c5b67ff (My-Project-Docker)
PublicIPs: 13.235.74.93 PrivateIPs: 172.31.7.207
```
- 2) mvn compile – to compile the source code.
  - 3) mvn test - to run the test of the project
  - 4) mvn package – to build and package the artifacts of the project to jar or war

mvn package from ec2 :: added war plugin in the pom.xml and verified the /target directory

```

INFO] [INFO] --- jacoco-maven-plugin:0.8.7:jacocoReport (jacoco-site) @ ABCtechnologies ---
INFO] Loading execution data file /home/ec2-user/edureka-project-1/target/jacoco.exec
INFO] Analyzed bundle 'RetailModule' with 2 classes
INFO]
INFO] BUILD SUCCESS
INFO] -----
INFO] total time: 10.244 s
INFO] Finished at: 2023-05-14T08:22:04Z
INFO] Final Memory: 13M/84M
INFO] [INFO] --
[ec2-user@ip-172-31-7-207 edureka-project-1]$ pwd
/home/ec2-user/edureka-project-1
[ec2-user@ip-172-31-7-207 edureka-project-1]$ cd target/
[ec2-user@ip-172-31-7-207 target]$ ls -lrt
total 6976
rwxrwxr-x 3 ec2-user ec2-user 35 May 14 08:18 maven-status
rwxrwxr-x 3 ec2-user ec2-user 25 May 14 08:18 generated-sources
rwxrwxr-x 3 ec2-user ec2-user 17 May 14 08:18 classes
rwxrwxr-x 3 ec2-user ec2-user 30 May 14 08:21 generated-test-sources
rwxrwxr-x 3 ec2-user ec2-user 13 May 14 08:21 test-classes
rwxrwxr-x 2 ec2-user ec2-user 113 May 14 08:22 generated-reports
rwxrwt-- 4 ec2-user ec2-user 4323 May 14 08:22 jacoco.exec
rwxrwxr-x 4 ec2-user ec2-user 54 May 14 08:22 ABCtechnologies-1.0
rwxrwxr-x 2 ec2-user ec2-user 7132823 May 14 08:22 ABCtechnologies-1.0.war
rwxrwxr-x 3 ec2-user ec2-user 20 May 14 08:22 site
[ec2-user@ip-172-31-7-207 target]$ ls [REDACTED]

```

i-0d25cc0f14c5b67ff (My-Project-Docker)  
PublicIPs: 13.235.74.93 PrivateIPs: 172.31.7.207



-----END of TASK 1-----

## Task 2: Set up the Git repository and push the source code. Then, log in to Jenkins.

1. Create a build pipeline containing a job for each
  - One for compiling source code
  - Second for testing source code
  - Third for packing the code
2. Execute the CI/CD pipeline to execute the jobs created in step 1
3. Set up a master-slave node to distribute the tasks in the pipeline.



## What is Jenkins?

Jenkins is an open-source automation server. It helps automate the parts of software development related to building, testing, and deploying, facilitating continuous integration and continuous delivery.

## How to install Jenkins?

```

sudo -i
sudo yum update -y
#Install Jenkins :
sudo wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo
sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
yum install jenkins

```

```

#Install epel Package:
amazon-linux-extras install epel
#Install Java:
amazon-linux-extras install java-openjdk11

#Start Jenkins:

Start jenkins service
service jenkins start

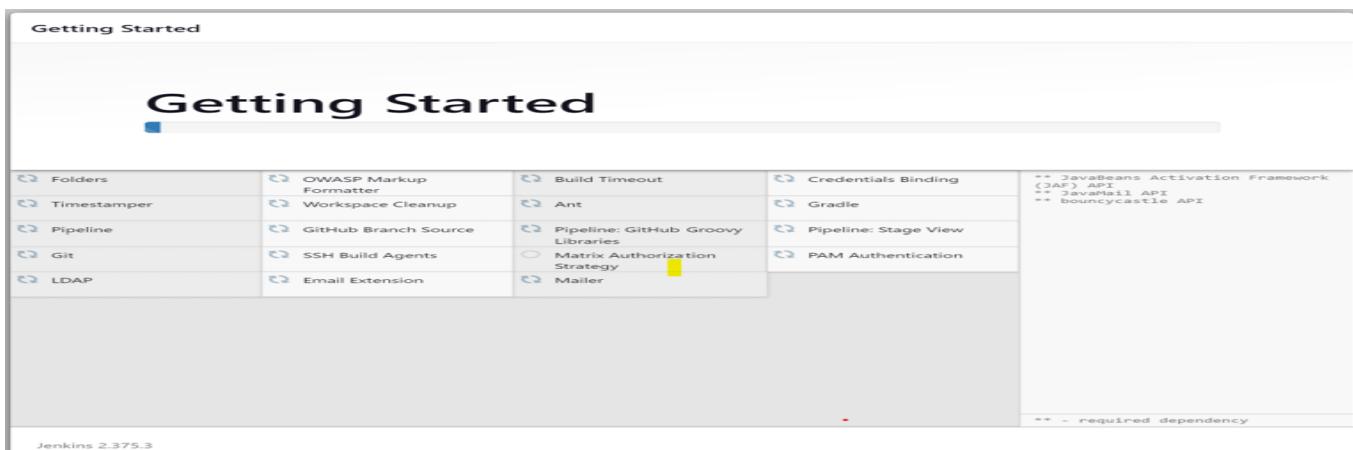
Setup Jenkins to start at boot,
chkconfig jenkins on

to get the admin password
cat /var/lib/jenkins/secrets/initialAdminPassword

```

Then the same can be accessed using the public IP : 8080 with admin password.

The screen will show a startup config, where you could choose to install the recommended plugins.



## JOB Creation- Source Code compilation

### **1)compiling source code 2) Test code and 3) Package code**

Steps :

- 1) Click New Item -> give a name to it Like -> compile Source code
- 2) Then click Freestyle project and save it.
- 3) We can then click on configure and add the github url from where the source code has to be pulled and then add the maven goals clean and compile. *I also faced error as my Jenkins was not able to pull the repo, so I went ahead and installed git in the EC2 machine to overcome the connectivity issue. [The details added in the document under GIT heading]*

Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

**Start building your software project**

- Create a job →
- Set up a distributed build
  - Set up on agent →
  - Configure a cloud →
  - Learn more about distributed builds →

**Build Queue**  
No builds in the queue.

**Build Executor Status**  
1 Idle  
2 Idle

## Adding Github details.

**Configure**

**General**

Description:  
This job will be used to compile the source code.  
[Plain text] [Preview]

Discard old builds ?  
 GitHub project  
Project url ?  
https://github.com/divyadevavasudevan/edureka-project-1

Display name ?  
edureka-project-1

This project is parameterized ?  
 Throttle builds ?  
 Execute concurrent builds if necessary ?

**Source Code Management**

Save Apply

## Adding Maven Goals – Compile goal is responsible to run mvn compile

**Configure**

**General**

Time-out actions ?  
Add action →

With Ant ?

**Build Steps**

Invoke top-level Maven targets ?  
Goals:  
clean compile  
Advanced...

Add build step ▾

**Post-build Actions**

Add post-build action ▾

Save Apply

```

Progress (1): 576/640 kB
Progress (1): 576/640 kB
Progress (1): 576/640 kB
Progress (1): 576/640 kB
Progress (1): 582/640 kB
Progress (1): 591/640 kB
Progress (1): 599/640 kB
Progress (1): 607/640 kB
Progress (1): 607/640 kB
Progress (1): 611/640 kB
Progress (1): 615/640 kB
Progress (1): 619/640 kB
Progress (1): 623/640 kB
Progress (1): 628/640 kB
Progress (1): 632/640 kB
Progress (1): 636/640 kB
Progress (1): 640 kB

Downloaded from central: https://repo.maven.apache.org/maven2/com/google/collections/google-collections/1.0/google-collections-1.0.jar (640 kB at 3.7 MB/s)

[INFO] Changes detected - recompiling the module!
[INFO] Compiling 3 source files to /var/lib/jenkins/workspace/compiling source code/target/classes
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 12.50s
[INFO] Finished at: 2023-02-25T08:42:28Z
[INFO] Final Memory: 23M/53M
[INFO] -----
Finished: SUCCESS

```



The Compiled source code can be viewed in the target directory of workspace.

**Workspace of compiling source code on Built-In Node**

- Status
- </> Changes
- Workspace
  - Wipe Out Current Workspace
- Build Now
- Configure
- Delete Project
- GitHub
- Rename

**Build History** trend ▾

| Build | Date                  | Status  |
|-------|-----------------------|---------|
| ② 42  | Feb 25, 2023, 8:42 AM | Success |
| ④ 41  | Feb 25, 2023, 8:40 AM | Failure |

Atom feed for all Atom feed for failures

Jenkins 2.375.3

## JOB 2 : Review and Test Code

The Github setup is same as the job 1.

The maven goal is verify and test – which is same as our mvn verify and mvn test commands

## Build Steps

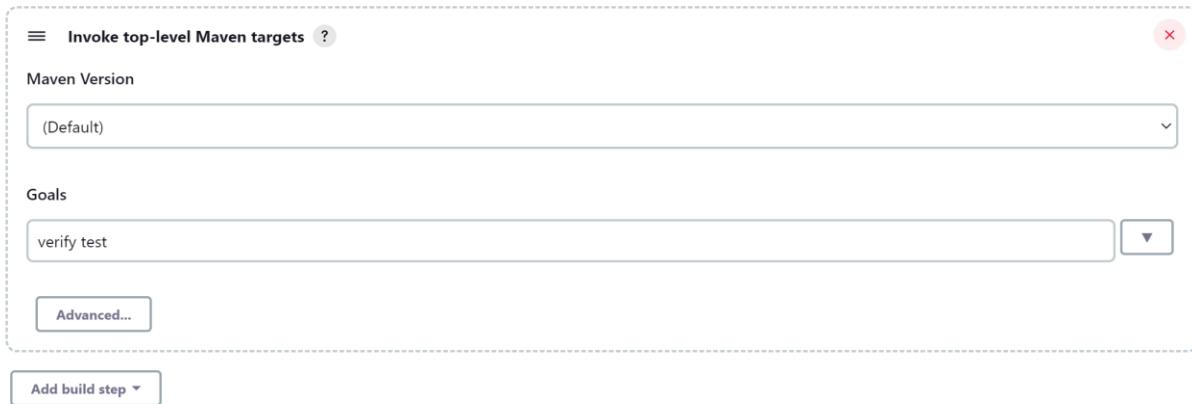
☰ Invoke top-level Maven targets ?

Maven Version  
(Default)

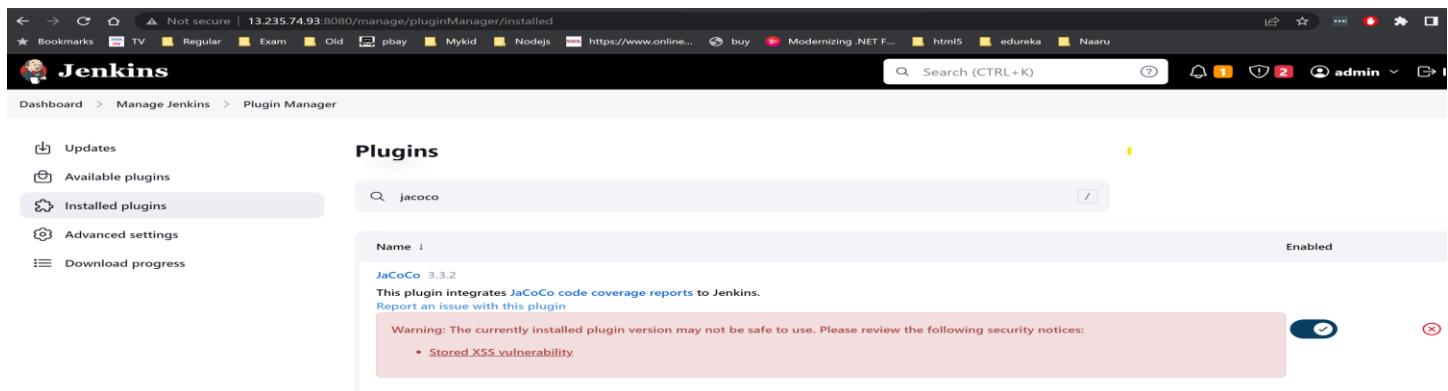
Goals  
verify test

Advanced...

Add build step ▾



Also configuring Jacoco reports in the post build step. We need the jacoco plugin



Updates Available plugins Installed plugins Advanced settings Download progress

Plugins

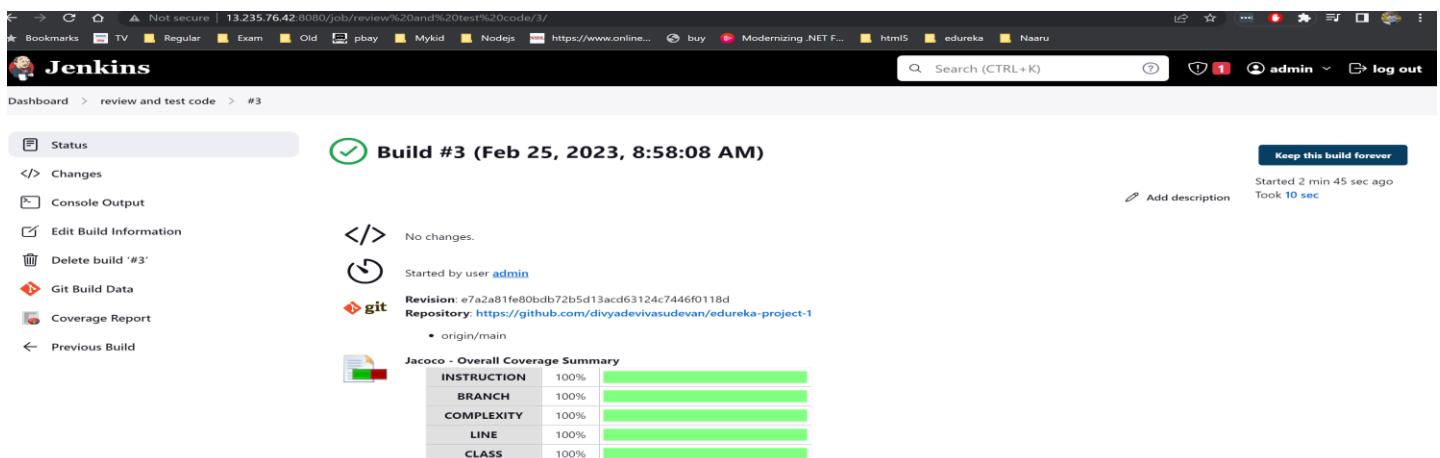
Name: JaCoCo 3.3.2 Enabled

This plugin integrates JaCoCo code coverage reports to Jenkins.

Report an issue with this plugin

Warning: The currently installed plugin version may not be safe to use. Please review the following security notices:

- Stored XSS vulnerability



Status Changes Console Output Edit Build Information Delete build '#3' Git Build Data Coverage Report Previous Build

Build #3 (Feb 25, 2023, 8:58:08 AM)

Keep this build forever

Started 2 min 45 sec ago Took 10 sec

Add description

</> No changes.

Started by user admin

Revision: e7a2a81f80bdb72b5d13acd63124c7446f0118d

Repository: <https://github.com/divyadevivasudevan/edureka-project-1>

origin/main

Jacoco - Overall Coverage Summary

|             | 100% |
|-------------|------|
| INSTRUCTION | 100% |
| BRANCH      | 100% |
| COMPLEXITY  | 100% |
| LINE        | 100% |
| CLASS       | 100% |



REST API Jenkins 2.375.3

compiling source c...zip ^ Show all ×

# JOB 3 : Package Code

Now package the code into a war file .

The following dependency should be added in the pom.xml if not there already

```
<plugin>
 <groupId>org.apache.maven.plugins</groupId>
 <artifactId>maven-war-plugin</artifactId>
 <version>3.3.1</version>
</plugin>
```

Then repeat the same procedure like above.

Create a new freestyle project and add Github connectivity.

Then in maven goals add package – this is equivalent to command mvn package

## Build Steps



Running this will ensure that war is created in the target folder. This can be used to deploy to our tomcat server and launch the application.

```
[INFO]
[INFO] --- maven-war-plugin:3.2.2:war (default-war) @ ABCtechnologies ---
[INFO] Packaging webapp
[INFO] Assembling webapp [ABCtechnologies] in [/var/lib/jenkins/workspace/package code/target/ABCtechnologies-1.0]
[INFO] Processing war project
[INFO] Copying webapp resources [/var/lib/jenkins/workspace/package code/src/main/webapp]
[INFO] Webapp assembled in [140 msecs]
[INFO] Building war: /var/lib/jenkins/workspace/package code/target/ABCtechnologies-1.0.war
[INFO]
[INFO] --- jacoco-maven-plugin:0.8.6:report (jacoco-site) @ ABCtechnologies ---
[INFO] Loading execution data file /var/lib/jenkins/workspace/package code/target/jacoco.exec
[INFO] Analyzed bundle 'RetailModule' with 2 classes
[INFO]
[INFO] --- maven-install-plugin:2.4:install (default-install) @ ABCtechnologies ---
[INFO] Installing /var/lib/jenkins/workspace/package code/target/ABCtechnologies-1.0.war to
/var/lib/jenkins/.m2/repository/com/abc/ABCtechnologies/1.0/ABCtechnologies-1.0.war
[INFO] Installing /var/lib/jenkins/workspace/package code/pom.xml to /var/lib/jenkins/.m2/repository/com/abc/ABCtechnologies/1.0/ABCtechnologies-1.0.pom
[INFO]
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 6.673 s
[INFO] Finished at: 2023-03-05T02:02:53Z
[INFO] Final Memory: 17M/84M
[INFO] -----
Finished: SUCCESS
```

## The Created all three jobs in a glance.

The screenshot shows the Jenkins dashboard with three pipeline jobs listed:

- compiling source code**: Last Success: 2 hr 46 min #3, Last Failure: N/A, Last Duration: 7.3 sec
- package code**: Last Success: 2 hr 46 min #2, Last Failure: N/A, Last Duration: 9.7 sec
- review and test code**: Last Success: 2 hr 46 min #4, Last Failure: N/A, Last Duration: 9.4 sec

Build Queue: No builds in the queue.

Build Executor Status: 1 idle, 2 idle.

The above three jobs can be integrated into a pipeline.

Create new item → Pipeline → provide the suitable name [I have given here CI Cd pipeline]  
The the commands to run the three tasks are coming from the pipeline script.

**Pipeline CI-CD-Pipeline-jobs**

**Stage View**

	Init	Compile code	Test Code	Package
Average stage times: (Average full run time: ~34s)	66ms	15s	8s	10s
#3 Mar 05 07:31	71ms	14s	17s	20s
#2 Mar 05 07:30	61ms	15s	52ms	56ms
#1 Mar 05 07:29				

**Permalinks**

Atom feed for all Atom feed for failures

The screenshot shows the Jenkins Pipeline configuration page. The pipeline script is defined as follows:

```

1 * pipeline {
2 agent any
3 stages {
4 stage('Init') {
5 steps {
6 echo "checkout git and compile"
7 }
8 }
9 stage('Compile code') {
10 steps {
11 build job: "compiling source code", wait:true
12 }
13 }
14 stage('Test Code') {
15 steps {
16 echo 'hi'
17 build job: "review and test code", wait: true
18 }
19 }
}

```

Below the script, there is a checkbox labeled "Use Groovy Sandbox". At the bottom are "Save" and "Apply" buttons.

## Master-Slave setup for jenkins

### Steps -

Installed Java on master node

Installed Jenkins,Java,Git,Maven on master node

Installed java,Git and Maven on slave node

Created a user and ssh keys on slave node

Copy keys on master node and join Slave with Master

Create Jenkins File – added slave labels to stages that should be run by slave.

### Jenkins Running on master::

```

root@ip-172-31-46-53 ~]# service jenkins start
Starting jenkins (via systemctl): [OK]
[root@ip-172-31-46-53 ~]# systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
 Loaded: loaded (/usr/lib/systemd/system/jenkins.service; disabled; vendor preset: disabled)
 Active: active (running) since Wed 2023-05-24 15:22:13 UTC; 44s ago
 Main PID: 4788 (java)
 Tasks: 44
 Memory: 410.7M
 CGroup: /system.slice/jenkins.service
 └─4788 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=%C/jenkins/war --httpPort=8080

```

Java installed in slave and created ssh key and user

```
[root@ip-172-31-42-183 ~]# useradd jsl1
[root@ip-172-31-42-183 ~]# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:TmVYU0y0MdgYdyK95dC9M7X6nI0B+wrxMYRBZIEI root@ip-172-31-42-183.ap-south-1.compute.internal
The key's randomart image is:
-----(RSA 2048)-----
+---[SHA256]-----+
[root@ip-172-31-42-183 ~]# cd .ssh/
[root@ip-172-31-42-183 .ssh]# cat id_rsa.pub > authorized_keys
[root@ip-172-31-42-183 .ssh]# chmod 700 authorized_keys
[root@ip-172-31-42-183 .ssh]# cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQDE90CfbgyAOflUzL1gyPq985Wkob5wu4uTWkzsYBJqWW8rzzOOwK5YCVYho3qjF/QrSu
MUaUPoinlps5oynV1Gxt5h2EreGDNvEtVzdQEFRuHrP3UpTaPKCQ1w6w907KRdSPufsUDBeTJZIx7GYtSBuDYJ3h6oUE51MdA1L
WXy23MvBPoO+LqEhnyoSIVtZ/6ATa/gGk/vl24GKuWyDfLnbxMjTEV/mqbOYohBfkZY99UAgV28H50PyZOCiE2keUrXKgbfRrhELiy
BCOW7ijdFrK4VXOwIHtmXDHS0JPl/+J27Dn/b/KvY1l8cQHbz02WWV2xUe/QCzpK4eQgx root@ip-172-31-42-183.ap-south-1.compute.internal
[root@ip-172-31-42-183 .ssh]#
```

i-015f412d01e322a6c (jenkins-slave)  
PublicIPs: 3.110.134.145 PrivateIPs: 172.31.42.183

## Copy keys in master

```
sudo mkdir /var/lib/jenkins/.ssh
sudo -i
```

```
chown jenkins:jenkins /var/lib/jenkins/.ssh/
echo "ssh-rsa
AAAAB3NzaC1yc2EAAAQABAAQDE90CfbgyAOflUzL1gyPq985Wkob5wu4uTWkzsYBJqWW8rzzOOwK5YCVYho3qjF/QrSu
MUaUPoinlps5oynV1Gxt5h2EreGDNvEtVzdQEFRuHrP3UpTaPKCQ1w6w907KRdSPufsUDBeTJZIx7GYtSBuDYJ3h6oUE51MdA1L
WXy23MvBPoO+LqEhnyoSIVtZ/6ATa/gGk/vl24GKuWyDfLnbxMjTEV/mqbOYohBfkZY99UAgV28H50PyZOCiE2keUrXKgbfRrhELiy
BCOW7ijdFrK4VXOwIHtmXDHS0JPl/+J27Dn/b/KvY1l8cQHbz02WWV2xUe/QCzpK4eQgx root@ip-172-31-42-183.ap-south-1.compute.internal" > /var/lib/jenkins/.ssh/known_hosts
```

```
cat /var/lib/jenkins/.ssh/known_hosts
```

```
ssh-rsa
```

```
AAAAB3NzaC1yc2EAAAQABAAQDE90CfbgyAOflUzL1gyPq985Wkob5wu4uTWkzsYBJqWW8rzzOOwK5YCVYho3qjF/QrSu
MUaUPoinlps5oynV1Gxt5h2EreGDNvEtVzdQEFRuHrP3UpTaPKCQ1w6w907KRdSPufsUDBeTJZIx7GYtSBuDYJ3h6oUE51MdA1L
WXy23MvBPoO+LqEhnyoSIVtZ/6ATa/gGk/vl24GKuWyDfLnbxMjTEV/mqbOYohBfkZY99UAgV28H50PyZOCiE2keUrXKgbfRrhELiy
BCOW7ijdFrK4VXOwIHtmXDHS0JPl/+J27Dn/b/KvY1l8cQHbz02WWV2xUe/QCzpK4eQgx root@ip-172-31-42-183.ap-south-1.compute.internal
```

```
https://aws.amazon.com/amazon-linux-2/
0 packages needed for security; 2 packages available
run "sudo yum update" to apply all updates.
ec2-user@ip-172-31-46-53 ~]$ sudo mkdir /var/lib/jenkins/.ssh
ec2-user@ip-172-31-46-53 ~]$ sudo -i
root@ip-172-31-46-53 ~]$ chown jenkins:jenkins /var/lib/jenkins/.ssh/
root@ip-172-31-46-53 ~]$ echo "ssh-rsa
AAAAB3NzaC1yc2EAAAQABAAQDE90CfbgyAOflUzL1gyPq985Wkob5wu4uTWkzsYBJqWW8rzzOOwK5YCVYho3qjF/QrSu
MUaUPoinlps5oynV1Gxt5h2EreGDNvEtVzdQEFRuHrP3UpTaPKCQ1w6w907KRdSPufsUDBeTJZIx7GYtSBuDYJ3h6oUE51MdA1L
WXy23MvBPoO+LqEhnyoSIVtZ/6ATa/gGk/vl24GKuWyDfLnbxMjTEV/mqbOYohBfkZY99UAgV28H50PyZOCiE2keUrXKgbfRrhELiy
BCOW7ijdFrK4VXOwIHtmXDHS0JPl/+J27Dn/b/KvY1l8cQHbz02WWV2xUe/QCzpK4eQgx root@ip-172-31-42-183.ap-south-1.compute.internal" > /var/lib/jenkins/.ssh/known_hosts
root@ip-172-31-46-53 ~]$ cat /var/lib/jenkins/.ssh/known_hosts
ssh-rsa
AAAAB3NzaC1yc2EAAAQABAAQDE90CfbgyAOflUzL1gyPq985Wkob5wu4uTWkzsYBJqWW8rzzOOwK5YCVYho3qjF/QrSu
MUaUPoinlps5oynV1Gxt5h2EreGDNvEtVzdQEFRuHrP3UpTaPKCQ1w6w907KRdSPufsUDBeTJZIx7GYtSBuDYJ3h6oUE51MdA1L
WXy23MvBPoO+LqEhnyoSIVtZ/6ATa/gGk/vl24GKuWyDfLnbxMjTEV/mqbOYohBfkZY99UAgV28H50PyZOCiE2keUrXKgbfRrhELiy
BCOW7ijdFrK4VXOwIHtmXDHS0JPl/+J27Dn/b/KvY1l8cQHbz02WWV2xUe/QCzpK4eQgx root@ip-172-31-42-183.ap-south-1.compute.internal
root@ip-172-31-46-53 ~]$ ssh-keyscan -H jenkins-slave.example.com >/var/lib/jenkins/.ssh/known_hosts
```

## 6. Join slave node to master

## To join the Jenkins slave node to Jenkins Master, performed below steps -

Select Build Executor Status > New Node > Type - Permanent

Name - jenkins-slave1

Description - jenkins-slave1

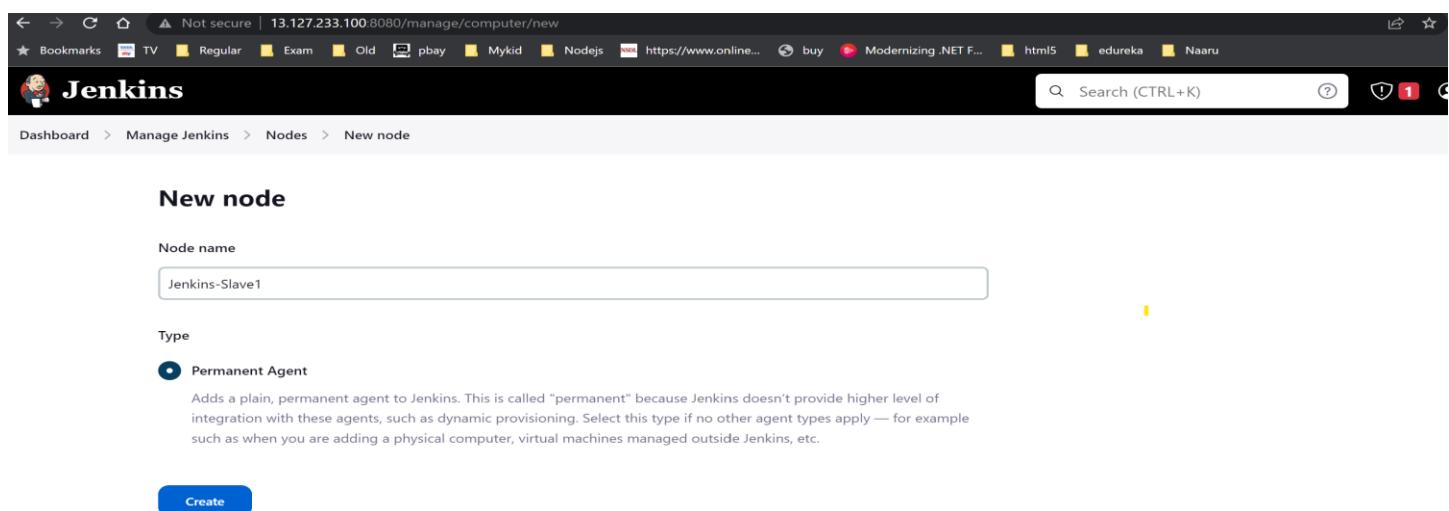
Number of executors - 1

Remote root directory - /home/ec2-user

Labels - jenkins-slave1

Usage - Use this mode as much as possible

Launch method - Launch agents via SSH



The screenshot shows the Jenkins 'New node' configuration page. At the top, there's a navigation bar with links like 'Dashboard', 'Manage Jenkins', 'Nodes', and 'New node'. Below the navigation, the title 'New node' is displayed. A 'Node name' field contains 'Jenkins-Slave1'. Under the 'Type' section, 'Permanent Agent' is selected. A descriptive text explains that this adds a plain, permanent agent to Jenkins. At the bottom, a blue 'Create' button is visible.

Gave the Host – IP and Credentials - use ssh username / private key options

Host Key Verification Strategy - Know hosts key strategy

Save and check that new slave node is getting connected. Once connected will display like below.

The screenshot shows the Jenkins web interface. At the top, there's a navigation bar with various links like Bookmarks, TV, Regular, Exam, Old, pbay, Mykid, Nodejs, https://www.online..., buy, Modernizing .NET F..., html5, edureka, and Nasaru. On the right side of the header, there are icons for a search bar, a user profile (admin), and a log out button.

The main content area is titled "Agent Jenkins-slave1". On the left, there's a sidebar with options: Status, Delete Agent, Configure, Build History, Load Statistics, Script Console, Log, System Information, and Disconnect. Below this is a section for "Build Executor Status" which shows 1 idle node.

In the center, it says "Agent is connected." and lists "Labels" (None). It also shows "Projects tied to Jenkins-slave1" with "js1" highlighted in blue. There's a link to "Add description" and a "Mark this node temporarily offline" button.



This screenshot shows the "Build Executor Status" page. It has a dropdown menu at the top. Below it, there are two sections: "Built-In Node" and "Jenkins-slave1".

- Built-In Node:** Shows 1 idle node.
- Jenkins-slave1:** Shows 1 idle node.

Executed the below pipeline script.

```
pipeline {
 agent {label 'js1'}

 stages {
 stage('checkout') {
 steps {
 git branch: 'main', url: 'https://github.com/divyadevivasudevan/edureka-project-1.git'
 }
 }
 stage('Execute Maven clean') {
 steps {
 sh 'mvn clean'
```

```

 }
 }

stage('Test code') {
 steps {
 sh 'mvn test verify'
 }
}

stage('Package Code') {
 steps {
 agent {label 'js1'}
 sh 'mvn package'
 }
}

```



The screenshot shows the Jenkins Pipeline Stage View for the 'Maven-app-CI-CD' pipeline. It displays the stages: 'checkout', 'Execute Maven clean', 'Test code', and 'Package Code'. The 'checkout' stage took 768ms, 'Execute Maven clean' took 326ms, 'Test code' took 317ms, and 'Package Code' took 318ms. The 'Test code' stage is highlighted in yellow. The pipeline is currently in a 'Master-slave' configuration. The sidebar shows various Jenkins management links like Status, Changes, Build Now, Configure, Delete Pipeline, Full Stage View, GitHub, Rename, Pipeline Syntax, Build History, and Filter builds.

checkout	Execute Maven clean	Test code	Package Code
768ms	326ms	317ms	318ms

Average stage times:  
(Average full run time: ~25s)

#20 May 25 13:18 No Changes

**Permalinks**

- Last build (#19), 6 min 9 sec ago
- Last failed build (#19), 6 min 9 sec ago
- Last unsuccessful build (#19), 6 min 9 sec ago

Committed the pipeline script to the repo. File name : JenkinsFile-masterslave.

----- END of TASK 2 -----

**Task 3:** Write a Docker file. Create an Image and container on the Docker host. Integrate docker host with Jenkins. Create CI/CD job on Jenkins to build and deploy on a container.

1. Enhance the package job created in step 1 of task 2 to create a docker image.
2. In the Docker image, add code to move the war file to the Tomcat server and build the image.



## What is docker?

Docker is a set of platforms as a service product that use OS-level virtualization to deliver software in packages called containers. The service has both free and premium tiers. The software that hosts the containers is called Docker Engine.

As pre-requisite :

Created a docker-hub account.

Installed docker in my VM.

The script for the same is committed with the file name :: installdocker.sh

```
sudo -i
sudo yum install -y yum-utils
sudo yum-config-manager --add-repo
https://download.docker.com/linux/rhel/9/x86_64/stable/repo/repodata/repomd.xml
sudo yum install docker
sudo systemctl start docker
sudo systemctl enable docker
```

## CI-CD pipeline creation.

1. Create new item → Pipeline → provide a name .

A screenshot of a web browser displaying the Jenkins 'New Item' creation dialog. The URL is '65.0.20.195:8080/view/all/newJob'. The page title is 'New Item [Jenkins]'. The main content area is titled 'Enter an item name' with a text input field containing 'Jenkins-docker-pipeline'. Below the input field, it says '= Required field'. There are four project types listed: 'Freestyle project' (with a brief description), 'Maven project' (with a brief description), 'Pipeline' (with a brief description), and 'Multi-configuration project' (with a brief description). The 'Pipeline' option is highlighted with a blue border.

2. Add Docker Hub access token into Jenkins
- Create new Access token on your DockerHub account under Security section.

The screenshot shows the Docker Hub security settings page for the user 'deeshuec2'. On the left, there's a sidebar with options like General, Security (which is selected), Default Privacy, Notifications, Convert Account, and Deactivate Account. The main area is titled 'Access Tokens' and contains a table with one row:

Description	Scope	Last Used	Created	Active
jenkins	Read, Write, Delete	Never	May 10, 2023 22:14:46	Yes

A blue button labeled 'New Access Token' is visible at the top right of the table area.

- Open Manage Jenkins → Mange Credentials and Add the System Credentials into Jenkins

## Manage Jenkins

New version of Jenkins (2.361.3) is available for download ([changelog](#)).

### System Configuration



#### Configure System

Configure global settings and paths.



#### Global Tool Configuration

Configure tools, their locations and automatic installers.

### Security



#### Configure Global Security

Secure Jenkins; define who is allowed to access/use the system.



#### Manage Credentials

Configure credentials

The screenshot shows the Jenkins 'Manage Jenkins' interface under the 'Credentials' section. It displays a table of stored credentials:

T	P	Store	Domain	ID	Name
		System	(global)	62d5afe6-862e-4946-92cf-720b50b8963c	jenkins
		System	(global)	8ad4b1db-0355-43e4-8f09-0d5e2cb19587	slave
		System	(global)	dockerhub	deeshuec2/***** (docker-hub)
		System	(global)	c8ab9663-154e-4ecc-9f12-0f90a7fad5f9	tomcat/*****

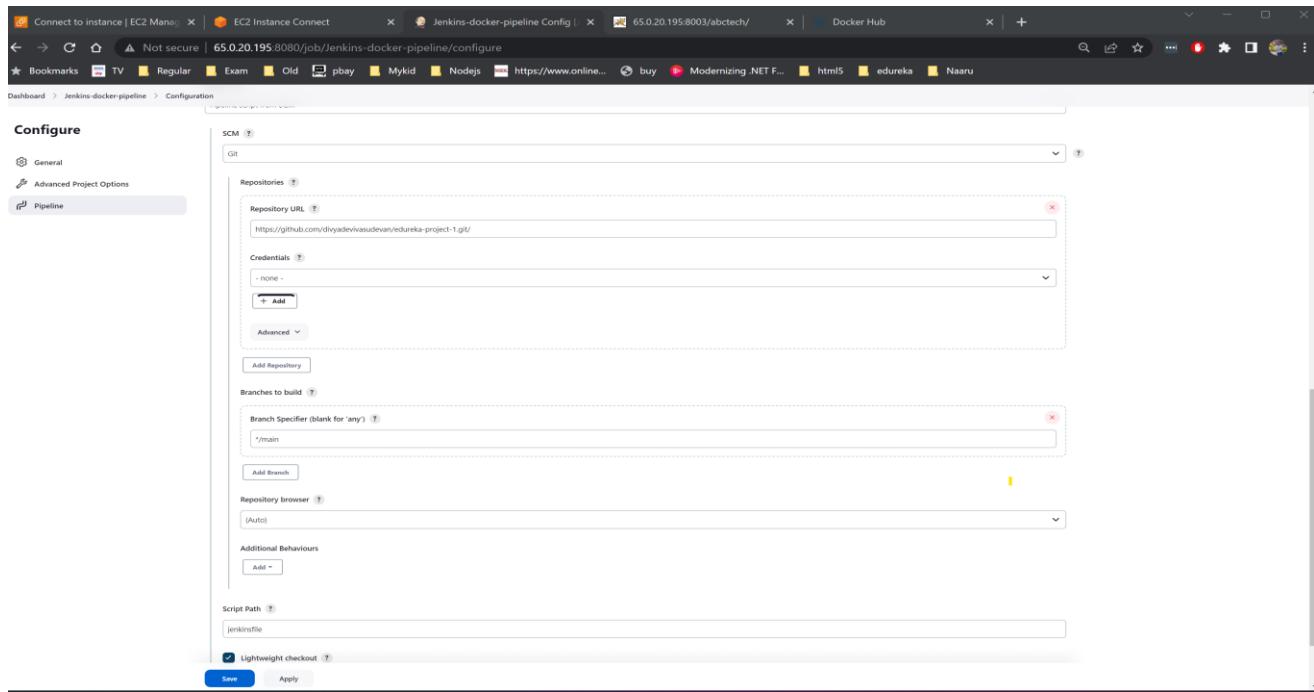
Below the table, there's a section titled 'Stores scoped to Jenkins' with a table showing a single entry:

P	Store	Domains
	System	(global)

At the bottom, there are icons for S, M, and L, and a note: 'Icon: S M L'.

The screenshot shows a Windows taskbar with several pinned icons, including File Explorer, Edge, and various system status indicators. The system tray shows the date and time as '19-05-2023 03:32 PM'. At the bottom right, there's a Jenkins status bar with the text 'REST API Jenkins 2.387.3' and a blue progress bar.

2. Go to Pipeline section add definition pipeline script from SCM and setup the other required config like below. Mainly the Jenkins filename / or path  
Branch to take from.



What is the Jenkins pipeline stages that we have written?

- We will checkout our git repo.
- Run maven clean install and package it to war.
- Run docker build to create a image and tag.
- Login to Docker hub and push the image.
- Start the container in 8003 port number.

While doing the above I faced errors:

- Was my jenkinsfilename I gave it wrong -> added a typo
- Login was not successful as I tried with DockerHub as the credential id, however I had it as dockerhub in the Jenkins credential id

These silly typos, made me to run the pipeline multiple times.

We Will also need a dockerfile that will do the following :

- Install tomcat
- Install java
- Copy our war file to the webapps
- And run the web server for us in the container

So we have now successfully used Jenkins and docker -> compiled , packaged the code and copied the war to tomcat and run the container and were able to access the web application running as a container using http://EC2-server-IP:8003/abctech/ - Code snippets and screenshot for these are in the below pages.

## Jenkinsfile

```
pipeline {
 agent any

 stages {
 stage('checkout') {
 steps {
 git branch: 'main', url: 'https://github.com/divyadevivasudevan/edureka-project-1.git'
 }
 }
 stage('Execute Maven') {
 steps {
 sh 'mvn clean package'
 sh 'echo package done'
 sh 'mv target/*.war target/abctech.war'
 }
 }
 }

 stage('Docker Build and Tag') {
 steps {
 sh 'docker build -t abctechapp:latest .'
 sh 'docker tag abctechapp deeshuec2/abctechapp:latest'
 }
 }

 stage('DockerHub Login and push image') {
 steps {
 //sh 'echo $DOCKERHUB_CREDENTIALS_PSW | docker login -u $DOCKERHUB_CREDENTIALS_USR --password-stdin'

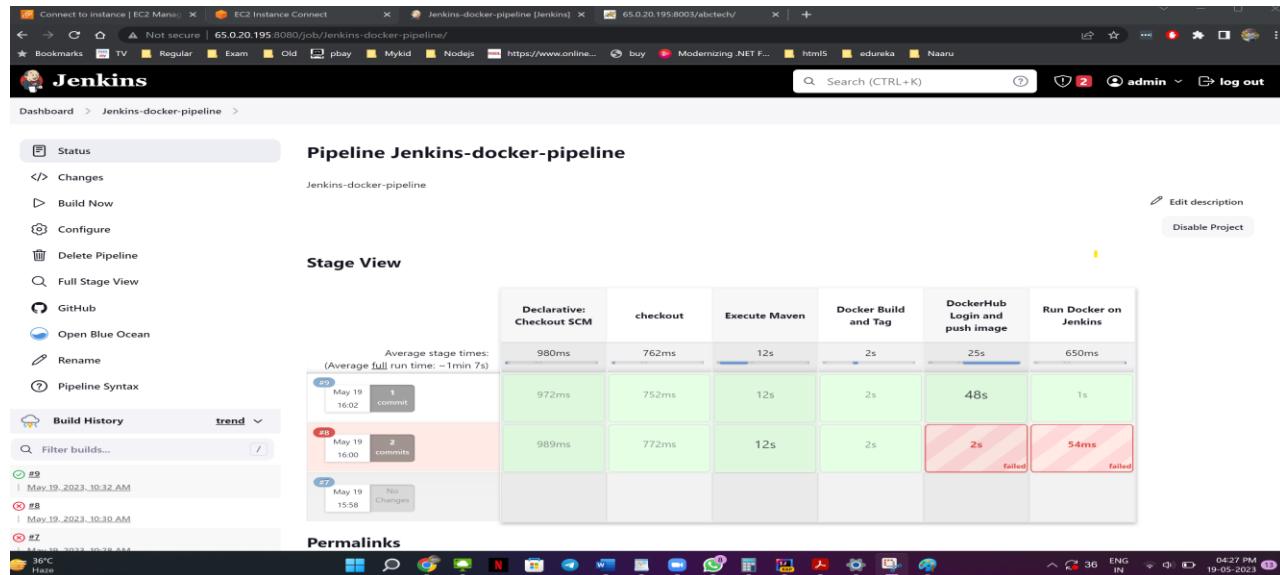
 withCredentials([usernamePassword(credentialsId: 'dockerhub',
 passwordVariable: 'dockerhubPassword', usernameVariable: 'dockerhubUser')]) {
 sh "docker login -u ${env.dockerHubUser} -p ${env.dockerHubPassword}"
 sh "docker push deeshuec2/abctechapp:latest"
 }
 }
 }

 stage('Run Docker on Jenkins') {
 steps {
 // sh "docker -H ssh://jenkins@172.31.28.25 run -d -p 8003:8080 deeshuec2/samplewebapp"
 sh "docker run -d -p 8003:8080 deeshuec2/abctechapp"
 }
 }
}
```

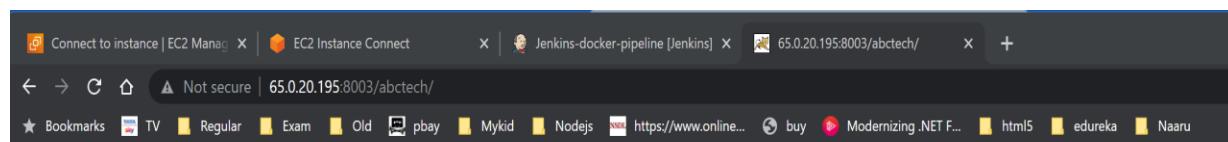
### 3. Add DockerFile with following commands.

```
FROM docker.io/library/ubuntu:18.04
RUN apt-get -y update && apt-get -y upgrade
RUN apt-get -y install openjdk-8-jdk wget
RUN mkdir /usr/local/tomcat
ADD https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.75/bin/apache-tomcat-9.0.75.tar.gz
/tmp/apache-tomcat-9.0.75.tar.gz
RUN cd /tmp && tar xvfz apache-tomcat-9.0.75.tar.gz
RUN cp -Rv /tmp/apache-tomcat-9.0.75/* /usr/local/tomcat/
ADD */*.war /usr/local/tomcat/webapps
EXPOSE 8089
CMD /usr/local/tomcat/bin/catalina.sh run
```

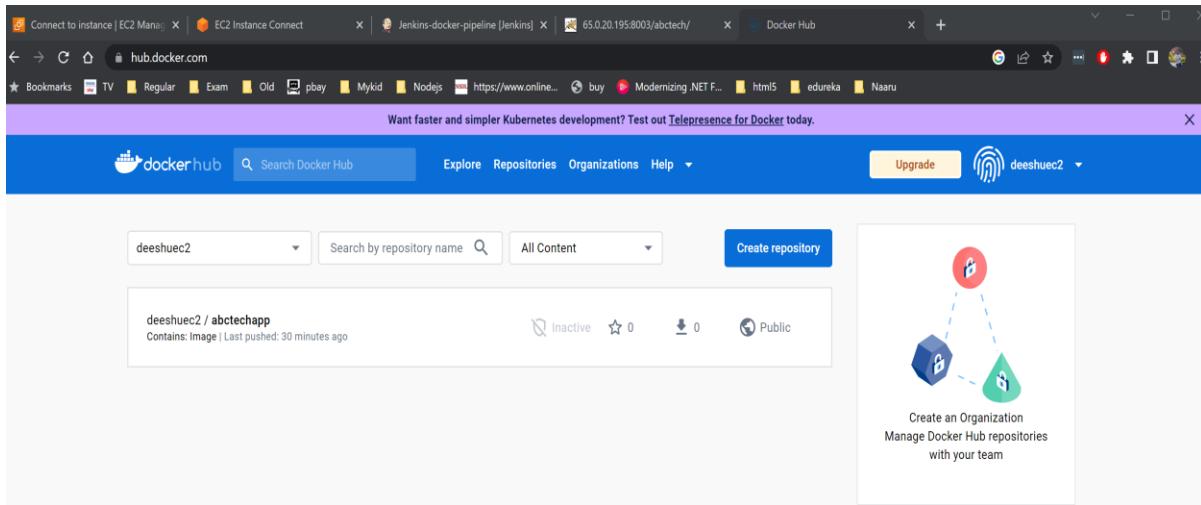
### 4. Screenshot of the various stages in build.



Access the web application running as a container using <http://EC2-server-IP:8003/abctech/> you should see the following web application.



We can also verify that our docker image was pushed to the docker hub.



---

-----END of TASK 3-----

**Task 4:** Integrate the Docker host with Ansible. Write an Ansible playbook to create an image and create a continuer. Integrate Ansible with Jenkins. Deploy Ansible-playbook. CI/CD job to build code on ansible and deploy it on docker container

1. Deploy Artifacts on Kubernetes
2. Write pod, service, and deployment manifest file
3. Integrate Kubernetes with Ansible
4. Ansible playbook to create deployment and service



What is ansible?

Ansible is a radically simple IT automation system. It handles configuration-management, application deployment, cloud provisioning, ad-hoc task-execution, and multinode orchestration - including trivializing things like zero-downtime rolling updates with load balancers.

Installing Ansible.

```
sudo yum install python
curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py
python3 get-pip.py --user
python3 -m pip install --user ansible
ansible --version
```

```

terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultExec Tunneling Packages Settings Help
Quick connect...
/home/ec2-user/
Name
.. .ssh .bash_logout .bash_profile .bashrc
2.6.2.170.09 (ec2-user)
100 2518k 100 2518k 0 0 24.1M 0 -:-:-- --:--:-- 24.1M
[root@ip-172-31-33-9 bin]# python3 get-pip.py --user
Collecting pip
 Downloading pip-23.1.2-py3-none-any.whl (2.1 MB)
 2.1/2.1 MB 30.4 MB/s eta 0:00:00
Collecting wheel
 Downloading wheel-0.40.0.whl (64 kB)
Successfully installed pip-23.1.2 wheel-0.40.0
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv
[root@ip-172-31-33-9 bin]# python3 -m pip install --user ansible
Collecting ansible
 Downloading ansible-7.5.0-py3-none-any.whl (43.6 MB)
 43.6/43.6 MB 12.4 MB/s eta 0:00:00
Collecting ansible-core==2.14.5 (from ansible)
 Downloading ansible_core-2.14.5-py3-none-any.whl (2.2 MB)
 2.2/2.2 MB 68.0 MB/s eta 0:00:00
Collecting jinja2>=3.0.0 (from ansible-core==2.14.5->ansible)
 Downloading Jinja2-3.1.2-py3-none-any.whl (133 kB)
 133.1/133.1 kB 14.3 MB/s eta 0:00:00
Requirement already satisfied: PyYAML>=5.1 in /usr/lib64/python3.9/site-packages (from ansible-core==2.14.5->ansible) (5.4.1)
Collecting cryptography (from ansible-core==2.14.5->ansible)
 Downloading cryptography-40.0.2-cp36abi3-manylinux_2_28_x86_64.whl (3.7 MB)
 3.7/3.7 MB 63.0 MB/s eta 0:00:00
Collecting packaging (from ansible-core==2.14.5->ansible)
 Downloading packaging-23.1-py3-none-any.whl (48 kB)
 48.9/48.9 kB 11.7 MB/s eta 0:00:00
Collecting resolvelib<0.9.0,>=0.5.3 (from ansible-core==2.14.5->ansible)
 Downloading resolvelib-0.8.1-py2.py3-none-any.whl (16 kB)
Collecting MarkupSafe>=2.0 (from jinja2>=3.0.0->ansible-core==2.14.5->ansible)
 Downloading MarkupSafe-2.1.2-cp39-cp39-manylinux_2_17_x86_64_manylinux2014_x86_64.whl (25 kB)
Collecting cffi>=1.12 (from cryptography->ansible-core==2.14.5->ansible)
 Downloading cffi-1.15.1-cp39-cp39-manylinux_2_17_x86_64_manylinux2014_x86_64.whl (441 kB)
 441.2/441.2 kB 55.1 MB/s eta 0:00:00
Collecting pycparser (from cffi>=1.12->cryptography->ansible-core==2.14.5->ansible)
 Downloading pycparser-2.21-py3.py3-none-any.whl (118 kB)
 118.7/118.7 kB 20.6 MB/s eta 0:00:00
Installing collected packages: resolvelib, pycparser, packaging, MarkupSafe, jinja2, cffi, cryptography, ansible-core, ansible
Successfully installed MarkupSafe-2.1.2 ansible-7.5.0 ansible-core-2.14.5 cffi-1.15.1 cryptography-40.0.2 jinja2-3.1.2 packaging-23.1 pycparser-2.21 resolvelib-0.8.1
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv
[root@ip-172-31-33-9 bin]#

```

UNREGISTERED VERSION - Please support Mobatek by subscribing to the professional edition here: <https://mobatext.mobatek.net>

30°C Mostly clear

ip-172-31-33-9.ap-south-1.compute.internal 1% 0.25 GB / 0.75 GB 0.01 Mb/s 0.00 Mb/s 14 min ec2-user /: 19% /boot: 31% /boot/efi: 1%

ENG IN WiFi 01:10 PM 20-05-2023

## Verifying the ansible version:

```

[ec2-user ~] $ curl -O https://pypa.github.io/wheel/wheel37/venv
[root@ip-172-31-33-9 bin]# ansible --version
ansible [core 2.14.5]
 config file = None
 configured module search path = ['/root/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
 ansible python module location = /root/.local/lib/python3.9/site-packages/ansible
 ansible collection location = /root/.ansible/collections:/usr/share/ansible/collections
 executable location = /root/.local/bin/ansible
 python version = 3.9.16 (main, Dec 8 2022, 00:00:00) [GCC 11.3.1 20221121 (Red Hat 11.3.1-4)] (/bin/python3)
 jinja version = 3.1.2
 libyaml = True
[root@ip-172-31-33-9 bin]#

```

## Integrating Docker host with Ansible

### Add Ansible Server user

```

$ sudo su -
$ useradd ansadmin
$ passwd ansadmin
Changing password for user ansadmin
New password:
Retype new password:
passwd: all authentication tokens updated successfully.

```

## Grant sudo Access to ansible user

```
$ visudo
```

```
ansadmin ALL=(ALL) NOPASSWD: ALL
```

## Setting up Password Authentication

```
$ vi /etc/ssh/sshd_config # To disable tunneled clear text passwords, change to no
here! PasswordAuthentication yes #PermitEmptyPasswords no #PasswordAuthentication no
```

## Creating SSHKEY

### Ssh-keygen

The screenshot shows a terminal session in Mobaxterm connected to an EC2 instance at 13.233.121.80. The terminal window title is "2. 13.233.121.80 (ec2-user)". The session content is as follows:

```
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ansadmin/.ssh/id_rsa):
Created directory '/home/ansadmin/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ansadmin/.ssh/id_rsa.
Your public key has been saved in /home/ansadmin/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:SxKPMVBewGvvaG/jmm/WzrDtyz5QViQCbR+ORA07X5w ansadmin@ip-172-31-7-207.ap-south-1.compute.internal
The key's randomart image is:
+---[RSA 2048]---+
| 0+=,+=== |
| .+000. |
| .o +.+ |
| ...Bo |
| . =E.S |
| . =,,+ + |
| .0* 0= |
| .,+*+.0 |
| |=+=0o. |
+---[SHA256]---+
[ansadmin@ip-172-31-7-207 root]$ cd /etc/ansible/
[ansadmin@ip-172-31-7-207 ansible]$ sudo vi hosts
[ansadmin@ip-172-31-7-207 ansible]$ ssh-copy-id localhost
/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/ansadmin/.ssh/id_rsa.pub"
The authenticity of host 'localhost (127.0.0.1)' can't be established.
ECDSA key fingerprint is SHA256:utqojm14eVnUbgKVSAh0bVgt5RpWY+Gpb/ZRC0.
ECDSA key fingerprint is MD5:b5:d5:f9:6a:b5:74:b9:c6:71:a5:13:42:48:31:69.
Are you sure you want to continue connecting (yes/no)? ansible all -m ping
Please type 'yes' or 'no': yes
/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
ansadmin@localhost's password:
Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'localhost'"
and check to make sure that only the key(s) you wanted were added.

[ansadmin@ip-172-31-7-207 ansible]$ ansible all -m ping
[WARNING]: Platform linux on host localhost is using the discovered Python interpreter at /usr/bin/python, but future installation of another Python interpreter could change this. See https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more information.
localhost | SUCCESS => {
 "ansible_facts": {
 "discovered_interpreter_python": "/usr/bin/python"
 },
 "changed": false,
 "ping": "pong"
}
[ansadmin@ip-172-31-7-207 ansible]$
```

The terminal window includes a file browser sidebar on the left and a system status bar at the bottom.

In Ansible server :: Create hosts file in /etc/ansible

```
$ sudo vi hosts
target server private-ip
localhost
Before the test localhost Keygen id copy
```

```
$ ssh-copy-id localhost
PING TEST
```

```
$ ansible all -m ping
```

```
localhost | SUCCESS => {
 "ansible_facts": {
 "discovered_interpreter_python": "/usr/bin/python"
 },
 "changed": false,
 "ping": "pong"
}
```

## Creating Ansible Playbook

```
$vi ansible-docker.yml
```

```

- hosts: localhost
 become: true
 tasks:
 - name: stop if we have old docker container
 command: docker stop devops-container
 ignore_errors: yes

 - name: remove stopped docker container
 command: docker rm devops-container
 ignore_errors: yes

 - name: remove current docker image
 command: docker rmi jayjodev/devops-image
 ignore_errors: yes

 - name: pull docker image from dockerhub
 command: docker pull deeshuec2/abctechapp:latest

 - name: creating docker image
 command: docker run -d --name devops-container -p 8010:8080 deeshuec2/abctechapp
```

```
[ansadmin@ip-172-31-7-207 docker]$ sudo vi ansible-docker-hub-pull-deploy.yml
[ansadmin@ip-172-31-7-207 docker]$ ansible-playbook -i hosts ansible-docker-hub-pull-deploy.yml
[WARNING]: Unable to parse /opt/docker/hosts as an inventory source
[WARNING]: No inventory was parsed, only implicit localhost is available
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'
PLAY [localhost] ****
TASK [Gathering Facts] ****
ok: [localhost]

TASK [stop if we have old docker container] ****
fatal: [localhost]: FAILED! => {"changed": true, "cmd": ["docker", "stop", "devops-container"], "delta": "0:00:00.090012", "end": "2023-05-21 08:15:27.656446", "msg": "non-zero return code", "rc": 1, "start": "2023-05-21 08:15:27.566434", "stderr": "Error response from daemon: No such container: devops-container", "stderr_lines": ["Error response from daemon: No such container: devops-container"], "stdout": "", "stdout_lines": []}
...ignoring

TASK [remove stopped docker container] ****
fatal: [localhost]: FAILED! => {"changed": true, "cmd": ["docker", "rm", "devops-container"], "delta": "0:00:00.065394", "end": "2023-05-21 08:15:27.987619", "msg": "non-zero return code", "rc": 1, "start": "2023-05-21 08:15:27.922225", "stderr": "Error: No such container: devops-container", "stderr_lines": ["Error: No such container: devops-container"], "stdout": "", "stdout_lines": []}
...ignoring

TASK [remove current docker image] ****
fatal: [localhost]: FAILED! => {"changed": true, "cmd": ["docker", "rmi", "jayjodev/devops-image"], "delta": "0:00:00.067644", "end": "2023-05-21 08:15:28.317993", "msg": "non-zero return code", "rc": 1, "start": "2023-05-21 08:15:28.250349", "stderr": "Error: No such image: jayjodev/devops-image", "stderr_lines": ["Error: No such image: jayjodev/devops-image"], "stdout": "", "stdout_lines": []}
...ignoring

TASK [pull docker image from dockerhub] ****
changed: [localhost]

TASK [creating docker image] ****
changed: [localhost]

PLAY RECAP ****
localhost : ok=6 changed=5 unreachable=0 failed=0 skipped=0 rescued=0 ignored=3
```

REPOSITORY TAG IMAGE ID CREATED SIZE  
abctech latest 5399986d5f6f 2 hours ago 560MB  
deeshuec2/abctechapp latest c97961eef911 42 hours ago 560MB  
abctechapp latest c97961eef911 42 hours ago 560MB  
deeshuec2/abctechapp <none> e330a037c3b5 46 hours ago 560MB  
<none> <none> e4028a00d387 46 hours ago 560MB  
<none> <none> 22620133d4d2 46 hours ago 560MB

## Verify Docker container

```
[ansadmin@ip-172-31-7-207 docker]$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
a7f86823502e deeshuec2/abctechapp "/bin/sh -c '/usr/lo..." 55 seconds ago Up 54 seconds 8089/tcp, 0.0.0.0:8010->8080/tcp, :::8010->8080/tcp devops-container
062ad6914109 abctechapp "/bin/sh -c '/usr/lo..." 9 minutes ago Up 9 minutes 8089/tcp, 0.0.0.0:8003->8080/tcp, :::8003->8080/tcp abc-retail-cart
[ansadmin@ip-172-31-7-207 docker]$
```

## Ansible and Jenkins Integration

Did this with the help of Ansible Plugin.

Name	Version
Ansible	1.0
Ansible Tower	0.11.1

We can see the Invoke Ansible Playbook option in the Build Environment section but we need to configure Ansible path for Jenkins.

Go to Manage Jenkins > Global Tool Configuration > It will display Ansible on the list

The screenshot shows the Jenkins Global Tool Configuration interface. Under the 'Ansible' section, there is a form to add a new Ansible installation. The 'Name' field is set to 'Ansible', and the 'Path to ansible executables directory' field is set to '/var/lib/jenkins/'. There is also a checkbox for 'Install automatically'. Below this, a list of existing Ansible installations is shown, with one entry named 'Ansible'. A red 'Delete Ansible' button is visible next to the list. Other sections like 'Docker' are partially visible at the bottom.

Then created a Freestyle project.  
Configured Ansible Plugin.

The screenshot shows the 'Build Steps' configuration for a Freestyle project. The 'Invoke Ansible Playbook' step is selected. In the 'Playbook path' field, 'ansible-docker-hub.yml' is specified. Under 'Inventory', the 'Do not specify Inventory' radio button is selected. The 'Host subset' field is empty. In the 'Credentials' section, 'jenkins' is chosen from a dropdown menu. The 'Vault Credentials' section shows a dropdown set to '- none -'. Under 'Privileges', the 'sudo' checkbox is checked, and 'sudo user' is set to 'ansadmin'. An 'Advanced' section is partially visible at the bottom.



## What is Kubernetes?

Kubernetes is an open-source container orchestration system for automating software deployment, scaling, and management. Google originally designed Kubernetes, but the Cloud Native Computing Foundation now maintains the project.

Deploy Artifacts on Kubernetes.

Applications can be installed in Kubernetes using Helm charts. Helm charts are packages that contain all the information that Kubernetes needs to know for managing a specific application within the cluster.

There are two different interfaces from which you can manage the resources on your cluster:

Kubernetes command line interface: kubectl

Kubernetes web-based user interface: Dashboard

## Installing Kubernetes

---

1. Download the latest release with the command:

```
2. curl -LO https://storage.googleapis.com/kubernetes-release/release/$(curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt)/bin/linux/amd64/kubectl
```

To download a specific version, replace the `$(curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt)` portion of the command with the specific version.

For example, to download version v1.7.0 on Linux, type:

```
curl -LO https://storage.googleapis.com/kubernetes-release/release/v1.7.0/bin/linux/amd64/kubectl
```

3. Make the kubectl binary executable.

```
4. chmod +x ./kubectl
```

5. Move the binary in to your PATH.

```
sudo mv ./kubectl /usr/local/bin/kubectl
```

```

← → ⌂ ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh?connType=standard&instanceId=i-0b36d7bb193f61ed9&osUser=ec2-user®ion=ap-south-1&sshPort=22#/
Bookmarks TV Regular Exam Old pbay Mykid Nodejs https://www.online... buy Modernizing .NET F... HTML5 edureka Naaru
AWS Services Search [Alt+S]
Last login: Tue May 23 03:11:46 2023 from ec2-13-233-177-4.ap-south-1.compute.amazonaws.com
[Amazon Linux 2 AMI]

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-37-37 ~]$ sudo minikube start
/bin/minikube: line 1: syntax error near unexpected token `<
/bin/minikube: line 1: <?xml version='1.0' encoding='UTF-8'?><Error><Code>NoSuchKey</Code><Message>The specified key does not exist.</Message><Details>No such object: minikube/releases/latest/minikube-OS_DISTRIBUTION- amd64</Details></Error>
[ec2-user@ip-172-31-37-37 ~]$ curl -LO https://storage.googleapis.com/kubernetes-release/release/$(curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt)
/bin/linux/amd64/kubectl
% Total % Received % Xferd Average Speed Time Time Current
 Dload Upload Total Spent Left Speed
100 46.9M 100 46.9M 0 0 66.8M 0:--:-- --:--:--:--:-- 66.9M
[ec2-user@ip-172-31-37-37 ~]$ curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt
-bash: v1.27.2: Command not found
[ec2-user@ip-172-31-37-37 ~]$ curl -s https://storage.googleapis.com/kubernetes-release/release/v1.7.0/bin/linux/amd64/kubectl
% Total % Received % Xferd Average Speed Time Time Current
 Dload Upload Total Spent Left Speed
100 69.0M 100 69.0M 0 0 15.8M 0:00:04 0:00:04 --:--:-- 16.6M
[ec2-user@ip-172-31-37-37 ~]$ chmod +x ./kubectl
[ec2-user@ip-172-31-37-37 ~]$ sudo mv ./kubectl /usr/local/bin/kubectl
[ec2-user@ip-172-31-37-37 ~]$ kubectl cluster-info
Kubernetes master is running at http://localhost:8080

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
[ec2-user@ip-172-31-37-37 ~]$

```

## Setting up Kubernetes Cluster

---

```

cat <<EOF > /etc/yum.repos.d/Kubernetes.repo
[Kubernetes]
name=Kubernetes
baseurl= https://packages.cloud.google.com/yum/repos/kubernetes-el7-x86_64
enabled=1
gpgcheck=1
repo_gpgcheck=0
gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg
https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg
EOF

```

```

cat <<EOF > /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-ip6tables=1
net.bridge.bridge-nf-call-iptables=1
EOF

```

sysctl --system

```

yum install -y kubeadm-1.21.3 kubelet-1.21.3 kubectl-1.21.3 --disableexcludes=kubernetes
yum install -y kubeadm-1.25 kubelet-1.25 kubectl-1.25 --disableexcludes=kubernetes

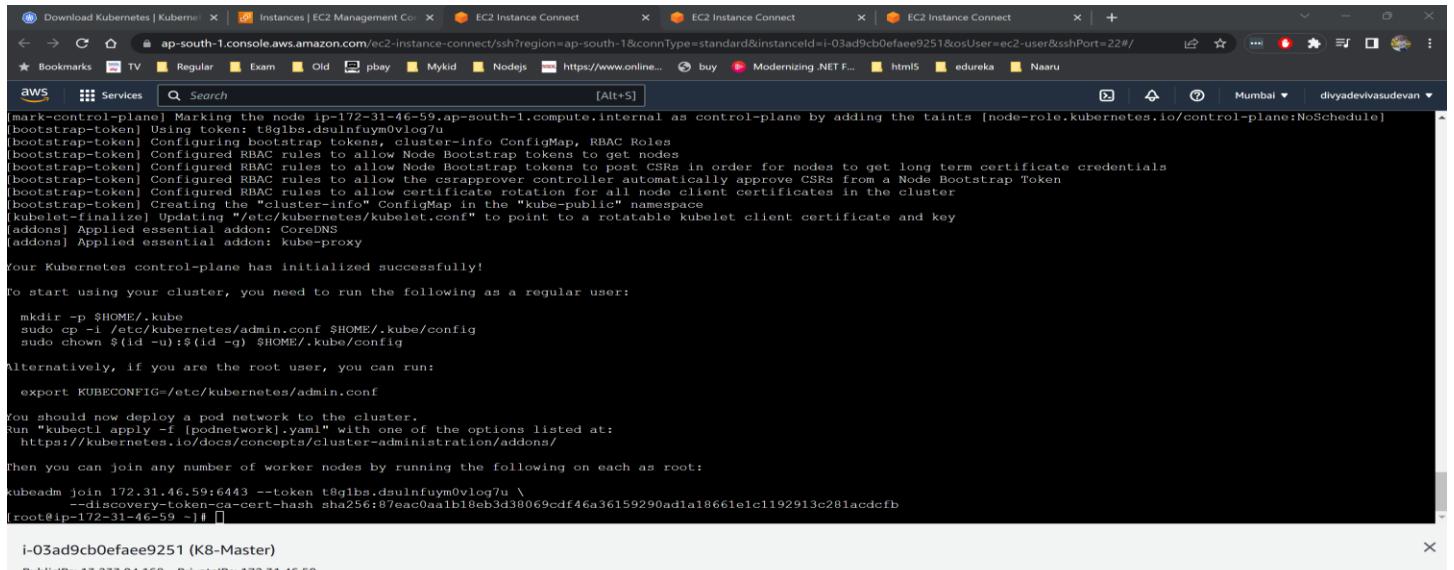
```

systemctl enable kubelet

```
systemctl start kubelet
```

```
sudo kubeadm init --apiserver-advertise-address=172.31.46.59 --pod-network-cidr=192.168.0.0/16 --ignore-preflight-errors=NumCPU --ignore-preflight-errors=Mem
```

```
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/
```



```
[mark-control-plane] Marking the node ip-172-31-46-59.ap-south-1.compute.internal as control-plane by adding the taints [node-role.kubernetes.io/control-plane:NoSchedule]
[bootstrap-token] Using token: t8q1bs.dsulfnyum0vlog7u
[bootstrap-token] Configuring bootstrap tokens, cluster-info ConfigMap, RBAC Roles
[bootstrap-token] Configured RBAC rules to allow Node Bootstrap tokens to get nodes
[bootstrap-token] Configured RBAC rules to allow Node Bootstrap tokens to post CSRs in order for nodes to get long term certificate credentials
[bootstrap-token] Configured RBAC rules to allow the csrapprover controller automatically approve CSRs from a Node Bootstrap Token
[bootstrap-token] Configured RBAC rules to allow certificate rotation for all node client certificates in the cluster
[kubelet-finalize] Creating the "cluster-info" ConfigMap in the "kube-public" namespace
[kubelet-finalize] Updating "/etc/kubernetes/kubelet.conf" to point to a rotatable kubelet client certificate and key
[addons] Applied essential addon: CoreDNS
[addons] Applied essential addon: kube-proxy

Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config

Alternatively, if you are the root user, you can run:

export KUBECONFIG=/etc/kubernetes/admin.conf

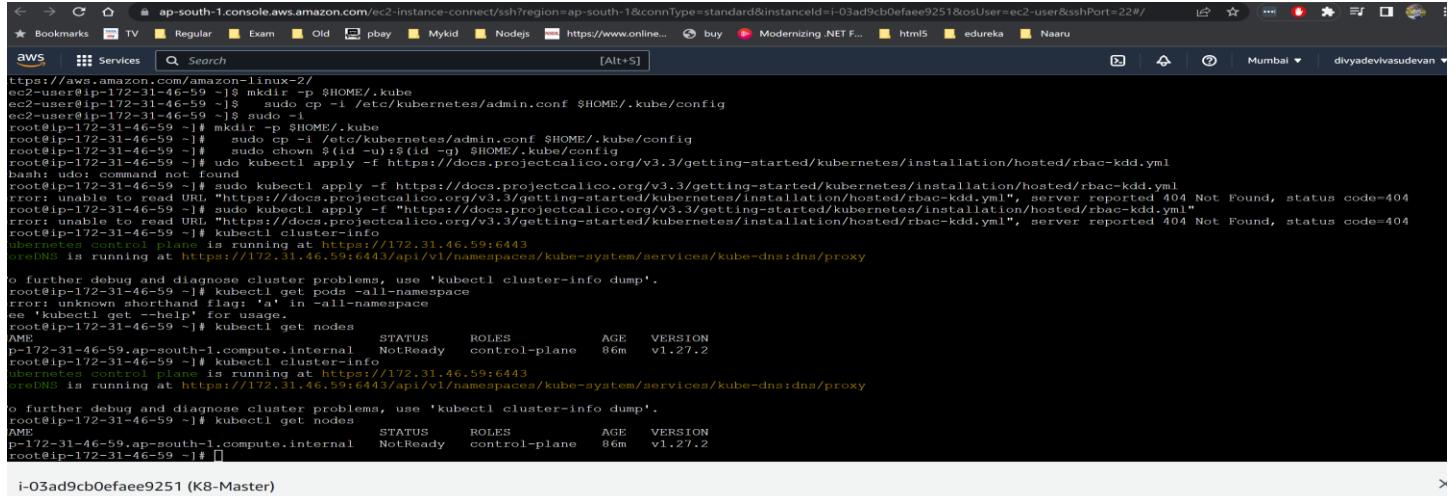
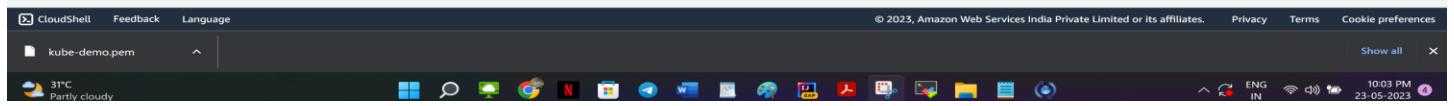
You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each as root:

kubeadm join 172.31.46.59:6443 --token t8q1bs.dsulfnyum0vlog7u \
--discovery-token-ca-cert-hash sha256:87eac0aalb18eb3d38069cdf46a36159290ad1a18661elc1192913c281acdcbf
root@ip-172-31-46-59 ~| #
```

i-03ad9cb0efaae9251 (K8-Master)

PublicIPs: 13.233.94.168 PrivateIPs: 172.31.46.59



```
https://aws.amazon.com/amazon-linux-2/
ec2-user@ip-172-31-46-59 ~| $ mkdir -p $HOME/.kube
ec2-user@ip-172-31-46-59 ~| $ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
ec2-user@ip-172-31-46-59 ~| $ sudo -i
root@ip-172-31-46-59 ~| # mkdir -p $HOME/.kube
root@ip-172-31-46-59 ~| # sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
root@ip-172-31-46-59 ~| # sudo chown $(id -u):$(id -g) $HOME/.kube/config
root@ip-172-31-46-59 ~| # sudo kubectl apply -f https://docs.projectcalico.org/v3.3/getting-started/kubernetes/installation/hosted/rbac-kdd.yaml
bash: sudo: command not found
root@ip-172-31-46-59 ~| # sudo kubectl apply -f https://docs.projectcalico.org/v3.3/getting-started/kubernetes/installation/hosted/rbac-kdd.yaml
error: unable to read URL "https://docs.projectcalico.org/v3.3/getting-started/kubernetes/installation/hosted/rbac-kdd.yaml", server reported 404 Not Found, status code=404
root@ip-172-31-46-59 ~| # sudo curl -s https://docs.projectcalico.org/v3.3/getting-started/kubernetes/installation/hosted/rbac-kdd.yaml
error: unable to read URL "https://docs.projectcalico.org/v3.3/getting-started/kubernetes/installation/hosted/rbac-kdd.yaml", server reported 404 Not Found, status code=404
root@ip-172-31-46-59 ~| # kubectl cluster-info
kubernetes control plane is running at https://172.31.46.59:6443
coreDNS is running at https://172.31.46.59:6443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

o further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
root@ip-172-31-46-59 ~| # kubectl get pods -all-namespaces
error: unknown shorthand flag: 'a' in -all-namespace
see 'kubectl get --help' for usage.
root@ip-172-31-46-59 ~| # kubectl get nodes
NAME STATUS ROLES AGE VERSION
p-172-31-46-59.ap-south-1.compute.internal NotReady control-plane 86m v1.27.2
root@ip-172-31-46-59 ~| # kubectl cluster-info
kubernetes control plane is running at https://172.31.46.59:6443
coreDNS is running at https://172.31.46.59:6443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

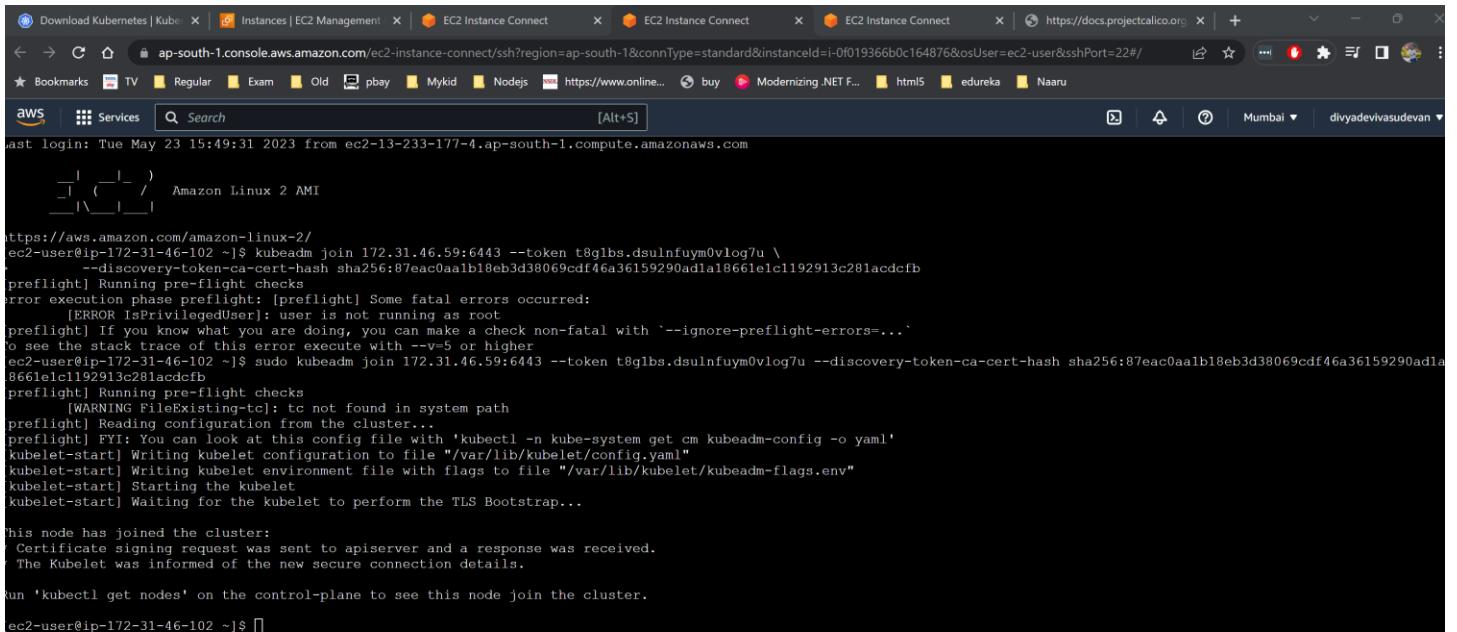
o further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
root@ip-172-31-46-59 ~| # kubectl get nodes
NAME STATUS ROLES AGE VERSION
p-172-31-46-59.ap-south-1.compute.internal NotReady control-plane 86m v1.27.2
root@ip-172-31-46-59 ~| #
```

i-03ad9cb0efaae9251 (K8-Master)

PublicIPs: 13.233.94.168 PrivateIPs: 172.31.46.59



Install the Kuberentes in the slave node as well and then use the token to join in the cluster.



```
Download Kubernetes | Kube | Instances | EC2 Management | EC2 Instance Connect | EC2 Instance Connect | EC2 Instance Connect | https://docs.projectcalico.org | +
← → C ⌂ ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=ap-south-1&connType=standard&instanceId=i-0f019366b0c164876&osUser=ec2-user&sshPort=22#/ Bookmarks TV Regular Exam Old pbay Mykid Nodejs https://www.online... buy Modernizing .NET F... html5 edureka Naaru
AWS Services Search [Alt+S]
last login: Tue May 23 15:49:31 2023 from ec2-13-233-177-4.ap-south-1.compute.amazonaws.com
_ _| (_ _) /Amazon Linux 2 AMI
_ \|_|_ |
https://aws.amazon.com/amazon-linux-2/
ec2-user@ip-172-31-46-102 ~]$ kubeadm join 172.31.46.59:6443 --token t8glbs.dsulnfuym0vlog7u \
--discovery-token-ca-cert-hash sha256:87eac0aalb18eb3d38069cdf46a36159290ad1a18661elc1192913c281acdcfb
preflight] Running pre-flight checks
error execution phase preflight: [preflight] Some fatal errors occurred:
[ERROR IsPrivilegedUser]: user is not running as root
[preflight] If you know what you are doing, you can make a check non-fatal with `--ignore-preflight-errors='...'
to see the stack trace of this error execute with --v=5 or higher
ec2-user@ip-172-31-46-102 ~]$ sudo kubeadm join 172.31.46.59:6443 --token t8glbs.dsulnfuym0vlog7u --discovery-token-ca-cert-hash sha256:87eac0aalb18eb3d38069cdf46a36159290ad1a18661elc1192913c281acdcfb
preflight] Running pre-flight checks
[WARNING FileExisting-ttc]: tc not found in system path
preflight] Reading configuration from the cluster...
preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubeadm-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubeadm-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubeadm-start] Starting the kubelet
[kubeadm-start] Waiting for the kubelet to perform the TLS Bootstrap...

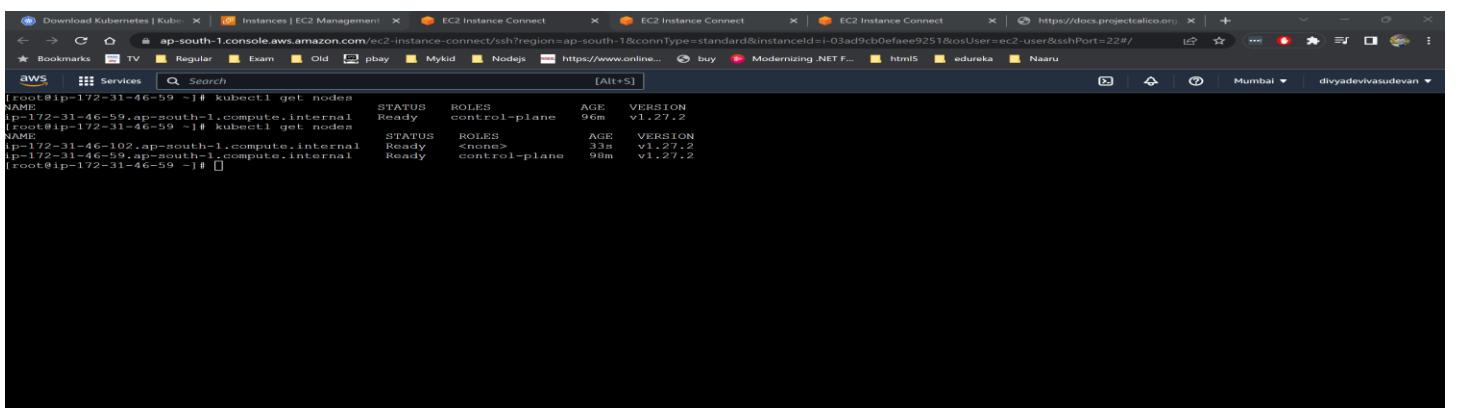
this node has joined the cluster:
 Certificate signing request was sent to apiserver and a response was received.
 The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.
ec2-user@ip-172-31-46-102 ~]$
```

i-0f019366b0c164876 (K8-Node1)  
PublicIPs: 13.235.128.172 PrivateIPs: 172.31.46.102



Control node and the worker node can now be seen.



```
Download Kubernetes | Kube | Instances | EC2 Management | EC2 Instance Connect | EC2 Instance Connect | EC2 Instance Connect | https://docs.projectcalico.org | +
← → C ⌂ ap-south-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=ap-south-1&connType=standard&instanceId=i-03ad9cb0efae9251&osUser=ec2-user&sshPort=22#/ Bookmarks TV Regular Exam Old pbay Mykid Nodejs https://www.online... buy Modernizing .NET F... html5 edureka Naaru
AWS Services Search [Alt+S]
root@ip-172-31-46-59 ~]$ kubectl get nodes
NAME STATUS ROLES AGE VERSION
ip-172-31-46-59.ap-south-1.compute.internal Ready control-plane 96m v1.27.2
ip-172-31-46-59.ap-south-1.compute.internal Ready <none> 33s v1.27.2
root@ip-172-31-46-59 ~]$
```

i-03ad9cb0efae9251 (K8-Master)  
PublicIPs: 13.233.94.168 PrivateIPs: 172.31.46.59



## Creating the Kubernetes manifest files for Deployment and service.

### deployment.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
 name: abctech-deployment
 labels:
 app: abctech
spec:
 replicas: 3
 selector:
 matchLabels:
 app: abctech
 template:
 metadata:
 labels:
 app: abctech
 spec:
 containers:
 - name: abctech
 image: deeshuec2/abctechapp:latest
 imagePullPolicy: Always
 ports:
 - containerPort: 8090
```

### app-service.yaml

```
apiVersion: v1
kind: Service
metadata:
 name: abctech-service
 labels:
 app: abctech
spec:
 type: LoadBalancer
 ports:
 - name: http
 port: 8082
 protocol: TCP
 targetPort: 8082
 selector:
 app: abctech
 sessionAffinity: None
```

### Kubectl commands ::

Kubectl get pods -> displays the list of pods/container running  
Kubectl cluster-info – gives details of the cluster

```
[root@ip-172-31-46-59 ~]# vi deployment.yaml
[root@ip-172-31-46-59 ~]# kubectl apply -f deployment.yaml
deployment.apps/abctech-deployment created
[root@ip-172-31-46-59 ~]# kubectl get pods
NAME READY STATUS RESTARTS AGE
abctech-deployment-748f4bdb66-mnxg4 0/1 ContainerCreating 0 7s
abctech-deployment-748f4bdb66-pq9vd 0/1 ContainerCreating 0 7s
abctech-deployment-748f4bdb66-z6zrz 0/1 ContainerCreating 0 7s
```

Access the application using cluster ip

The screenshot shows a Firefox browser window. The address bar displays '10.111.143.7:8082/abctech/'. The main content area of the browser shows a web page with the following text:

**Welcome to ABC technologies**

**This is retail portal**

**Add Product** **View Product**

Ansible Playbook for K8 deployment

```

- hosts: all
 become: true
 tasks:
 - name: Create Production namesapce
 k8s:
 name: production
 api_version: v1
 kind: Namespace
 state: present
 - name: Create new deployment
 command: kubectl apply -f deployment.yaml
 - name: Create new service
 command: kubectl apply -f app-service.yaml
```

---

----- END of Task 4 -----

**Task 5:** Using Prometheus, monitor the resources like CPU utilization: Total Usage, Usage per core, usage breakdown, memory, and network on the instance by providing the endpoints on the local host. Install the node exporter and add the URL to the target in Prometheus.

Using this data, log in to Grafana and create a dashboard to show the metrics

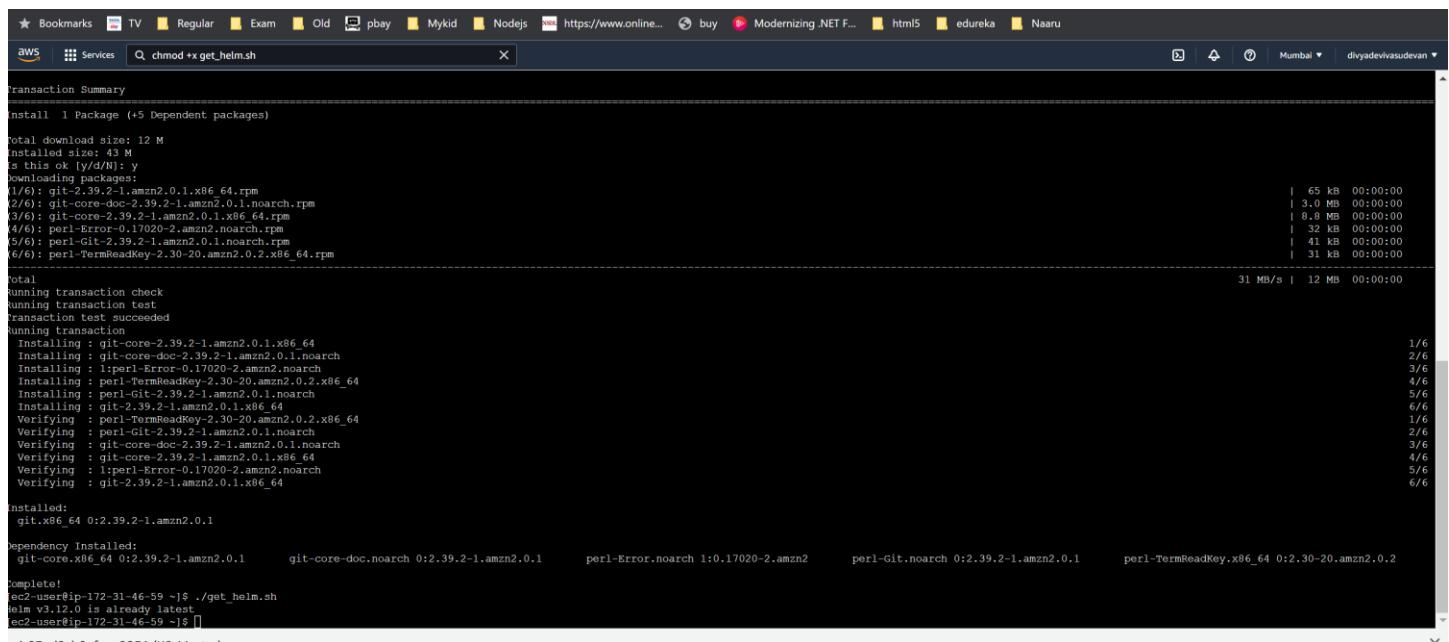
**Prometheus:** An open-source monitoring system with a dimensional data model, flexible query language, efficient time series database and modern alerting approach.

**Grafana:** Grafana is a multi-platform open-source analytics and interactive visualization web application. It provides charts, graphs, and alerts for the web when connected to supported data sources

## Monitoring using Prometheus and Grafana

Installing Helm .

```
curl -fsSL -o get_helm.sh https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3
chmod +x get_helm.sh
./get_helm.sh
```



```
curl -fsSL -o get_helm.sh https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3
chmod +x get_helm.sh
./get_helm.sh

transaction Summary
=====
install 1 Package (+5 Dependent packages)

total download size: 12 M
installed size: 43 M
is this ok [y/N]: y
downloading packages:
1/6) git-core-2.39.2-1.amzn2.0.1.x86_64.rpm | 65 kB 00:00:00
2/6) git-core-doc-2.39.2-1.amzn2.0.1.noarch.rpm | 3.0 MB 00:00:00
3/6) git-core-2.39.2-1.amzn2.0.1.x86_64.rpm | 8.8 MB 00:00:00
4/6) perl-Error-0.17020-2.amzn2.noarch.rpm | 32 kB 00:00:00
5/6) perl-Git-2.39.2-1.amzn2.0.1.noarch.rpm | 41 kB 00:00:00
6/6) perl-TermReadkey-2.30-20.amzn2.0.2.x86_64.rpm | 31 kB 00:00:00
31 MB/s | 12 MB 00:00:00

total
running transaction check
running transaction test
transaction test succeeded
running transaction
installing : git-core-2.39.2-1.amzn2.0.1.x86_64 | 1/6
installing : git-core-doc-2.39.2-1.amzn2.0.1.noarch | 2/6
installing : perl-Error-0.17020-2.amzn2.noarch | 3/6
installing : perl-Git-2.39.2-1.amzn2.0.1.noarch | 4/6
installing : perl-TermReadkey-2.30-20.amzn2.0.2.x86_64 | 5/6
installing : perl-Git-2.39.2-1.amzn2.0.1.noarch | 6/6
verifying : perl-Git-2.39.2-1.amzn2.0.1.noarch | 1/6
verifying : git-core-doc-2.39.2-1.amzn2.0.1.noarch | 2/6
verifying : git-core-2.39.2-1.amzn2.0.1.x86_64 | 3/6
verifying : perl-TermReadkey-2.30-20.amzn2.0.2.x86_64 | 4/6
verifying : perl-Error-0.17020-2.amzn2.noarch | 5/6
verifying : perl-Git-2.39.2-1.amzn2.0.1.noarch | 6/6

installed:
git.x86_64 0:2.39.2-1.amzn2.0.1

dependency Installed:
git-core.x86_64 0:2.39.2-1.amzn2.0.1 git-core-doc.noarch 0:2.39.2-1.amzn2.0.1 perl-Error.noarch 1:0.17020-2.amzn2.0.1 perl-Git.noarch 0:2.39.2-1.amzn2.0.1 perl-TermReadkey.x86_64 0:2.30-20.amzn2.0.2

complete!
[root@ip-172-31-46-59 ~]# ./get_helm.sh
helm v3.17.0 is already latest!
[root@ip-172-31-46-59 ~]#
```

## Installed Prometheus

```
helm repo add prometheus-community https://prometheus-community.github.io/helm-charts
helm install prometheus prometheus-community/prometheus
```



```
root@ip-172-31-46-59 ~]# helm repo add prometheus-community https://prometheuscommunity.
bash: helm: command not found
root@ip-172-31-46-59 ~]# helm repo add prometheus-community https://prometheuscommunity-community.github.io/helm-charts
bash: helm: command not found
root@ip-172-31-46-59 ~]# /usr/local/bin/helm repo add prometheus-community https://prometheuscommunity-community.github.io/helm-charts
error looks like "https://prometheuscommunity-community.github.io/helm-charts" is not a valid chart repository or cannot be reached: failed to fetch https://prometheuscommunity-community.github.io/helm-charts/index.yaml : 404 Not Found
root@ip-172-31-46-59 ~]# /usr/local/bin/helm repo add prometheus-community https://prometheus-community.github.io/helm-charts
prometheus-community" has been added to your repositories
root@ip-172-31-46-59 ~]#
```

```

[...]
error: command not found
root@ip-172-31-46-59 ~]# /usr/local/bin/heilm repo add prometheus-community https://prometheuscommunity-community.github.io/helm-charts
error: looks like "https://prometheuscommunity-community.github.io/helm-charts" is not a valid chart repository or cannot be reached: failed to fetch https://prometheuscommunity-community.git
Not Found
[...]
root@ip-172-31-46-59 ~]# /usr/local/bin/heilm repo add prometheus-community https://prometheus-community.github.io/helm-charts
"prometheus-community" has been added to your repositories
root@ip-172-31-46-59 ~]# /usr/local/bin/heilm install prometheus prometheus-community/prometheus
NAME: prometheus
LAST DEPLOYED: Fri May 26 16:17:39 2023
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
The Prometheus server can be accessed via port 80 on the following DNS name from within your cluster:
prometheus-server.default.svc.cluster.local

Get the Prometheus server URL by running these commands in the same shell:
export POD_NAME=$(kubectl get pods --namespace default -l "app=prometheus,component=server" -o jsonpath=".items[0].metadata.name")
kubectl --namespace default port-forward $POD_NAME 9090

The Prometheus alertmanager can be accessed via port 9093 on the following DNS name from within your cluster:
prometheus-%!s(<nil>).default.svc.cluster.local

Get the Alertmanager URL by running these commands in the same shell:
export POD_NAME=$(kubectl get pods --namespace default -l "app=prometheus,component=" -o jsonpath=".items[0].metadata.name")
kubectl --namespace default port-forward $POD_NAME 9093
#####
WARNING: Pod Security Policy has been disabled by default since k8s 1.25+. use
it deprecated after k8s 1.25+. use
(index .Values "prometheus-node-exporter" "rbac"
"pspEnabled" with (index .Values
"prometheus-node-exporter" "rbac" "pspAnnotations")
in case you still need it.
#####

The Prometheus PushGateway can be accessed via port 9091 on the following DNS name from within your cluster:
prometheus-prometheus-pushgateway.default.svc.cluster.local

```

Get the PushGateway URL by running these commands in the same shell:

[kubectl port-forward -n prometheus deploy/prometheus-server 8080:9090](#)

Dashboard [Jenkins]    Prometheu... Settings - G X    Problem loading page    Prometheu... Time Series X +

← → ⌛ 10.109.96.20:9090/graph

Getting Started    Dashboard [Jenkins]    Apache Tomcat/9.0...

Prometheus    Alerts    Graph    Status    Help

Enable query history

Expression (press Shift+Enter for newlines)

Execute - insert metric at cursor - ↴

Graph    Console

◀ Moment ▶

Element

no data

Add Graph

## Install Grafana

helm repo add grafana <https://grafana.github.io/helm-charts>

```
[root@ip-172-31-46-59 ~]# helm repo add grafana https://grafana.github.io/helm-charts
-bash: helm: command not found
[root@ip-172-31-46-59 ~]# /usr/local/bin/helm repo add grafana https://grafana.github.io/helm-charts
"grafana" has been added to your repositories
[root@ip-172-31-46-59 ~]#
```

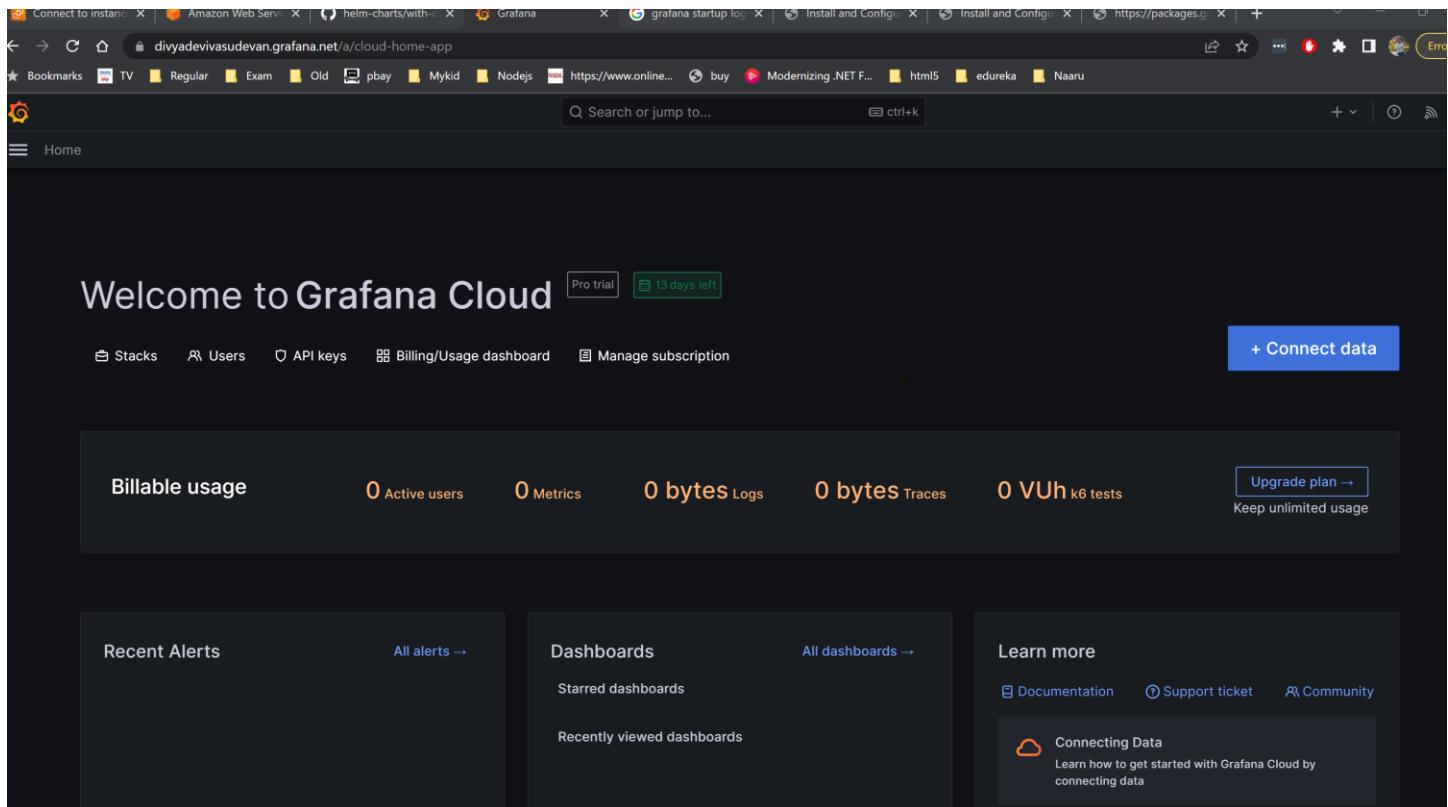
helm install my-release grafana/grafana

```
Error: INSTALLATION FAILED: Kubernetes cluster unreachable: error loading config file "/home/ec2-user/.kube/config": open /home/ec2-user/.kube/config: permission denied
[ec2-user@ip-172-31-46-59 ~]$ /usr/local/bin/helm install my-release grafana
[ec2-user@ip-172-31-46-59 ~]$ sudo /usr/local/bin/helm install my-release grafana
"grafana" has been added to your repositories
[ec2-user@ip-172-31-46-59 ~]#
```

LAST DEPLOYED: Sat May 27 03:51:13 2023  
NAMESPACE: default  
STATUS: deployed  
REVISION: 1  
NOTES:  
1. Get your 'admin' user password by running:  
  kubectl get secret --namespace default my-release-grafana -o jsonpath=".data.admin-password" | base64 --decode ; echo  
  
2. The Grafana server can be accessed via port 80 on the following DNS name from within your cluster:  
  my-release-grafana.default.svc.cluster.local  
  
  Get the Grafana URL to visit by running these commands in the same shell:  
  export POD\_NAME=\$(kubectl get pods --namespace default -l "app.kubernetes.io/name=grafana,app.kubernetes.io/instance=my-release" -o jsonpath=".items[0].metadata.name")  
  kubectl --namespace default port-forward \$POD\_NAME 3000  
  
3. Login with the password from step 1 and the username: admin  
  \*\*\*\*\* WARNING: Persistence is disabled!!! You will lose your data when \*\*\*\*\*  
  \*\*\*\*\* the Grafana pod is terminated.\*\*\*\*\*  
[ec2-user@ip-172-31-46-59 ~]\$

My Grafana wasn't starting up properly.

So I used Grafana cloud instance.



The screenshot shows a web browser window with the following details:

- Address Bar:** Shows the URL [@cloud-home-app](https://divyadevivasudevan.grafana.net).
- Header:** Includes a search bar labeled "Search or jump to..." and a "ctrl+k" keyboard shortcut.
- Welcome Section:** Displays "Welcome to Grafana Cloud" with a "Pro trial" button and a "13 days left" badge.
- Top Navigation:** Features links for "Stacks", "Users", "API keys", "Billing/Usage dashboard", "Manage subscription", and a prominent blue "Connect data" button.
- Metrics Section:** Shows billable usage statistics: 0 Active users, 0 Metrics, 0 bytes Logs, 0 bytes Traces, 0 VUh k6 tests, and a "Upgrade plan" button with the subtext "Keep unlimited usage".
- Bottom Left:** A "Recent Alerts" card with a link to "All alerts →".
- Bottom Middle:** A "Dashboards" card with links to "Starred dashboards" and "Recently viewed dashboards".
- Bottom Right:** A "Learn more" card with links to "Documentation", "Support ticket", "Community", and a "Connecting Data" section with a subtext about getting started with Grafana Cloud by connecting data.

## Created a role for integration AWS cloud watch

Stack name

Stack name  
GrafanaLabs-CloudWatch-Metrics

Parameters

GrafanaLabs Account

AccountID  
This is Grafana Lab's AWS account ID and is used to allow Grafana Cloud access to your AWS metrics.  
arn:aws:iam::008923505280:root

ExternalID  
This is your Grafana Cloud identifier and is used for security purposes.  
1009139

New Role

Role name  
Customize the name of the IAM role used by GrafanaLabs for the CloudWatch Integration.  
GrafanaLabsCloudWatchIntegration

Capabilities

The following resource(s) require capabilities: [AWS::IAM::Role]  
This template contains Identity and Access Management (IAM) resources. Check that you want to create each of these resources and that they have the minimum required permissions. In addition, they have custom names. Check that the custom names are unique within your AWS account. [Learn more](#)

I acknowledge that AWS CloudFormation might create IAM resources with custom names.

CloudShell Feedback Language © 2023, Amazon Web Services India Private Limited or its affiliates. Privacy Terms Cookie preferences

## 2. Connect to AWS account

Once you have successfully created a new AWS IAM role, you can proceed with the installation by entering your account info below.

### ARN

Paste the ARN from your AWS IAM role.

arn:aws:iam::869405457603:role/GrafanaLabsCloudWatchIntegration

### AWS Regions

ap-south-1 X

**Test Connection**

✓ All good! The account is working properly.

### 3. Create scrape job

Select which services you want scraped. Dashboards will be automatically installed for all supported services.

 New AWS services available to scrape!

Scrape job name

Monitoring

Services to scrape

AWS/ApplicationELB × ec2 ×

Include your AWS resource tags on an aws\_<service-name>\_info metric (ex, aws\_ec2\_info)



Tags will appear as labels on the exported metric with a `tag_` prefix. Choosing to include tags will increase the total number of active series which can have an impact on your Grafana Cloud Costs. Additionally, please ensure your tags adhere to AWS best practices in that they do not contain personally identifiable information (PII) or other confidential or sensitive information.

**Create scrape job**

✓ All good! Scrape job created.

Import dashboard from Grafana.com

### Integration - CloudWatch Metrics

Manage folder dashboards and permissions

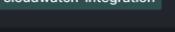
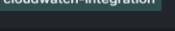
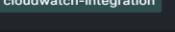
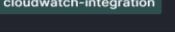
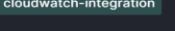
 Dashboards  Panels  Alert rules  Permissions  Settings

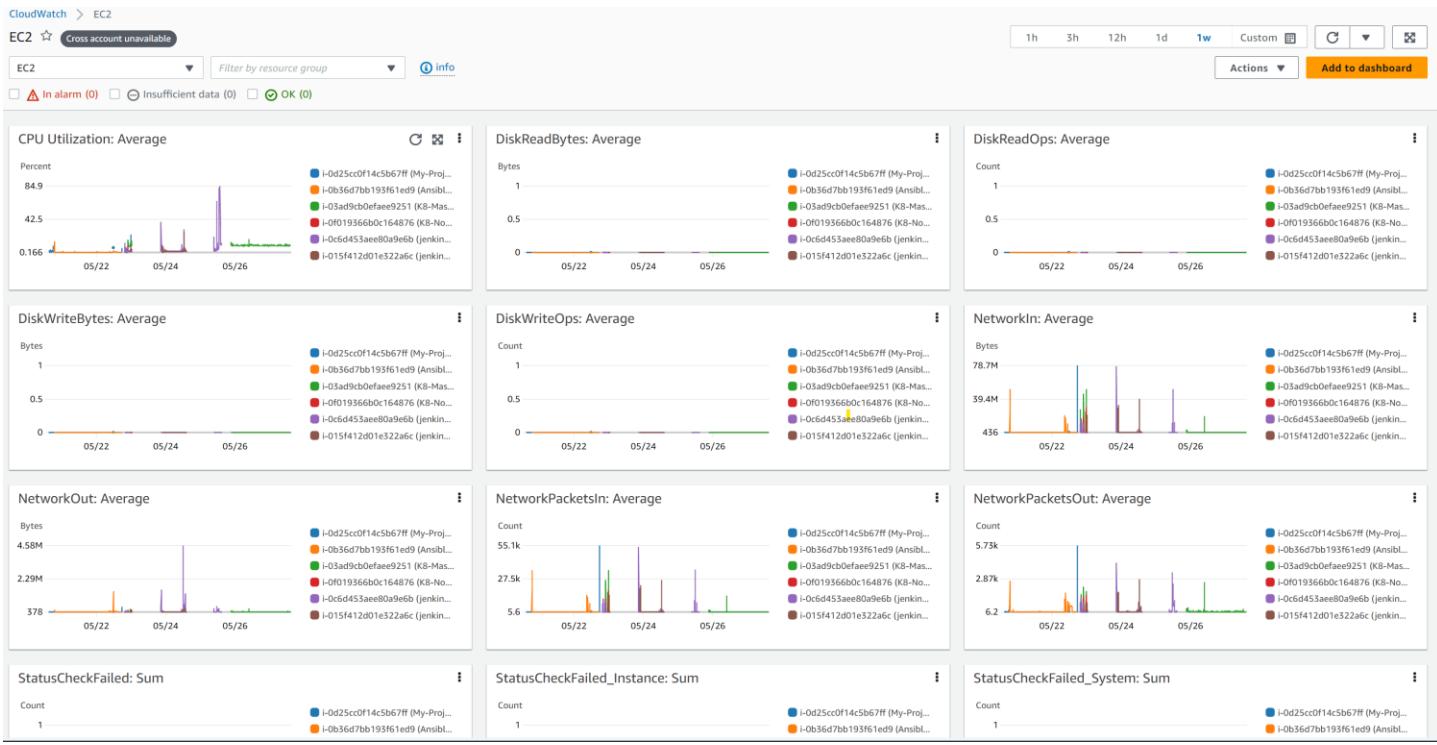
Search for dashboards

New ▾

 Filter by tag  Starred

 Sort

<input type="checkbox"/>	<b>AWS EBS</b>	 Integration - CloudWatch Metrics	
<input type="checkbox"/>	<b>AWS EC2</b>	 Integration - CloudWatch Metrics	
<input type="checkbox"/>	<b>AWS ECS</b>	 Integration - CloudWatch Metrics	
<input type="checkbox"/>	<b>AWS Lambda</b>	 Integration - CloudWatch Metrics	
<input type="checkbox"/>	<b>AWS RDS</b>	 Integration - CloudWatch Metrics	
<input type="checkbox"/>	<b>AWS S3</b>	 Integration - CloudWatch Metrics	



-----END of Project-----